Competitive subset selection with two agents

Gaia Nicosia [a] Andrea Pacifici [b] Ulrich Pferschy [c]

[a] *Dipartimento di Informatica e Automazione, Università degli Studi "Roma Tre", Italy. Email:* nicosia@dia.uniroma3.it

[b] *Dipartimento di Ingegneria dell'Impresa, Università degli Studi di Roma "Tor Vergata", Italy, Email:* pacifici@disp.uniroma2.it

[c] *Institut für Statistik und Operations Research, Universität Graz, Austria, Email:* pferschy@uni-graz.at

Abstract

We address an optimization problem in which two agents, each with a set of weighted items, compete in order to maximize the total weight of their *winning sets*. The latter are built according to a sequential game consisting in a fixed number of rounds. In every round each agent submits one item for possible inclusion in its winning set. We study two natural rules to decide the winner of each round.

For both rules we deal with the problem from different perspectives. From a centralized point of view, we investigate (*i*) the structure and the number of efficient (i.e. Pareto optimal) solutions, (*ii*) the complexity of finding such solutions, (*iii*) the best-worst ratio, i.e. the ratio between the efficient solution with largest and smallest total weight, and (*iv*) existence of Nash equilibria.

Finally, we address the problem from a single agent perspective. We consider *preventive* or *maximin* strategies, optimizing the objective of the agent in the worst case, and *best response* strategies, where the items submitted by the other agent are known in advance either in each round (on-line) or for the whole game (off-line).

Key words: multi-agent optimization, combinatorial game theory, on-line strategies, computational complexity

1 Introduction

We consider a multi-agent problem where agents compete to fill a joint solution set with their items. We focus on the following situation: There are two agents, each of them owning one of two disjoint sets of weighted items. The agents have to select items from their set for putting them in a solution set. This process proceeds in a fixed number of rounds. In every round each of the

two agents selects exactly one of its items and submits the item for possible inclusion in the solution set. A central decision mechanism chooses one of the items as "winner" of this round. The winning item is permanently included in the solution set. We consider two versions of the problem, depending whether the losing item is permanently discarded or can be reused in the succeeding rounds. Each agent wants to maximize its total solution value which is given by the total weight of its items included in the solution set. We assume complete information, i.e. both agents know all items' weights.

1.1 Related literature

The problem we address can be regarded as a single-suit card game in which each of two players chooses a card from its hand. The highest value card wins and each card can be used only once. In [11] the authors study a zero-sum game in which the cards are submitted simultaneously and the players want to maximize the total value of the won cards. In [8,12] the so called *whistette* game is addressed. There is a totally ordered suit of $2n$ cards, distributed between the two players. The player who has the lead plays first on each trick. The player with the higher card wins a trick and obtains the lead. Players want to maximize the number of won tricks.

Moreover, seeds assignment in team sport tournaments, e.g. chess leagues, is indeed related to the problem we deal with in this paper. The players of each team are ordered and each player faces the opponent on the same ordered position of the other teams' list. However, the ordering of players may be restricted to obey an established ranking (e.g. ELO points) to some extent. There the objective is to maximize the number of wins while our problem aims at the maximization of the total weight of all winning rounds.

Another problem strictly related to ours and the above described seeds assignment is addressed in [9], where the authors investigate optimal strategies on how to choose a player (item) for the next match (round) in a game consisting of a sequence of matches. Two types of games are considered, given a winning probability for every pair of competing players. In the first type, after each match, the loser is eliminated from the list of remaining players while the winner remains in the list. In the second type, both players are eliminated after each match.

In addition, our problem can be viewed as a special knapsack game where two agents try to fit their items in a bounded common solution set in order to maximize their profits. In our case, we have a unit size for the items and a special mechanism to decide which items fit, i.e. are accepted in the common knapsack. Although the problem that we address in this work is relatively

new, 0–1 knapsack problems (KP) in a multi-decision environment have been considered in the literature for two decades: from game-theoretic to auction applications there is a variety of papers dealing with this classical combinatorial optimization problem. Hereafter, we limit to report a few of them.

A related problem in which different players try to fit their own items in a common knapsack is the so called *knapsack sharing problem* studied by several authors (see for instance [6,7]). A single objective function that tries to balance the profits among the players is considered in a centralized perspective. Another interesting game, based on the maximum 0–1 knapsack, interpreted as a special on-line problem, is addressed in [10] where a two person zero-sum game, called *knapsack game*, is considered. Knapsack problems are also addressed in the context of auctions. For instance, in [1], an application for selling advertisements on Internet search engines is considered. In particular, there are n agents wishing to place an item in the knapsack and each agent gives a private valuation for having an item in the knapsack, while each item has a publicly known size. Finally, in [3], a two-agent knapsack problem where one agent (the leader) controls the capacity of a knapsack, and the other (follower) solves the resulting knapsack problem, is tackled by dynamic programming techniques. Note that the case of a single agent was extensively treated in [5].

1.2 Formal Problem Setting

In the following, A resp. B indicate the agents' names each of them owning a set of n items, where item i has weight a_i resp. b_i. Throughout this paper we assume the items to be sorted in decreasing order of weights, i.e. $a_1 \geq a_2 \geq \ldots \geq a_n$ resp. $b_1 \geq b_2 \geq \ldots \geq b_n$. Sometimes we will identify items by their weight. All information about the input list of items is public.

The game is performed over c rounds. In each round both of the two agents simultaneously submit one of their items, say a_i and b_j. We consider the two most natural rules for deciding which of the two submitted items wins and is added to the solution set.

Rule 1 (R1): if $a_i \geq b_j$ then A wins;

Rule 2 (R2): if $a_i \leq b_j$ then A wins [1].

Moreover, under both rules, we deal with two different scenarios.

Single-use items: each losing item is discarded and can not be submitted a second time;

[1] In case of a tie we assume that A always wins.

Reusable items: each losing item can be reused for submission in the succeeding rounds.

In conclusion, we tackle four different versions of the competitive subset selection problem (CSS), which we denote as $CSS(\alpha, \beta)$, where $\alpha \in \{R1, R2\}$ depending on the rule in force and $\beta \in \{\text{single}, \text{reuse}\}$ depending on the treatment of the losing items.

This problem can be represented by a graph model. Each agent's item is associated to a node of a weighted complete bipartite graph $G = (V^A \cup V^B, E^A \dot{\cup} E^B)$. An arc (i, j) belongs to E^A or to E^B depending on the winner of a comparison of a_i and b_j, which of course depends on the applied selection rule.

Rule R1: arc (i, j) with weight $w_{ij} = \max\{a_i, b_j\}$ belongs to E^A if $a_i \geq b_j$, i.e. if A wins, otherwise it belongs to E^B;
Rule R2: arc (i, j) with weight $w_{ij} = \min\{a_i, b_j\}$ belongs to E^A if $a_i \leq b_j$, i.e. if A wins, otherwise it belongs to E^B;

Every pair of items (a_i, b_j) submitted simultaneously in one round can be represented by an arc (i, j) between the two corresponding nodes. Hence, any solution may be represented as a set M of c arcs on G. The total weight of items in the solution for agent A is given by $w^A(M) = \sum_{ij \in M \cap E^A} w_{ij}$ and that of B by $w^B(M) = \sum_{ij \in M \cap E^B} w_{ij}$.

Note that, in case of single-use items, the set M is a matching on G of cardinality c. Thus, determining a global optimum maximizing the sum of the two agents' weights can be done in polynomial time by solving a weighted cardinality assignment problem [4]. Yet, when losing items are reusable, M may not be a matching, however it is easy to describe the set of feasible solutions using a linear program, whose set of constraints is a proper subset of the constraints used for the single-use items case.

Indeed, in Section 2, we show that the global optimum for all versions of CSS can be found in a straightforward way, without recurring to matching techniques or linear programming.

1.3 Our results

The problem we deal with in this paper is viewed from different perspectives. We first consider it from a centralized point of view as a bicriteria optimization problem and study the structure of efficient, i.e. Pareto optimal (PO) solutions (Section 2). We give bounds on the ratio between the efficient so-

lutions with largest and smallest total weight (Section 2.2) and characterize the computational complexity of finding one and/or all Pareto optimal solutions (Section 2.1). Moreover, we discuss the existence of Nash equilibria (Section 2.3).

In addition, in Section 3, we address the problem from a single agent perspective. We consider the problem of optimizing the objective of one agent when *partial* information on the sequence of items submitted by the other is given. We consider two different scenarios: In the first one, the sequence of the second agent is completely unknown, and one seeks for strategies that maximize the weight of the first agent in the worst case (Section 3.1). In game theory literature, this is often referred to as a *maximin strategy*. In the second scenario, the items submitted by the other agent are known in advance in each round and we propose *best response* strategies for the first agent (Section 3.2).

Our results are summarized in Table 1. Although most of the results for the reusable items scenario are similar to those of the single-use items case and can be shown by slightly modifying the arguments, there are a few interesting differences between the two scenarios.

Problem type	PO recogn.	# PO sol.	Nash-Eq.	Best-Worst Ratio	MaxiMin	Best resp.
$CSS(R1, single)$	poly	$O(c)$	no	2	trivial	2-apx
$CSS(R1, reuse)$	poly	$O(c)$	yes	unbounded	algor.	2-apx
$CSS(R2, single)$	NP-hard	$\Omega(2^c)$	no	unbounded	algor.	opt
$CSS(R2, reuse)$	poly	$O(c)$	open	unbounded if $n>c$ c if $n=c$	algor.	open

Table 1
Table of results.

2 Efficient solutions and equilibria

In this section we investigate the solution structure in an offline perspective, i.e. we consider the problem from a centralized and static point of view and look for Pareto efficient solutions. Then, we show that under both rules, no Nash equilibria exist, in general.

2.1 Pareto Efficient Solutions

Using the same terminology as in multicriteria optimization, a solution M is called *Pareto efficient* or simply *efficient* or *nondominated* if there exist no solution M' such that $w^A(M) \leq w^A(M')$, $w^B(M) \leq w^B(M')$ and $w^A(M) + w^B(M) < w^A(M') + w^B(M')$.

We start our investigation with Rule 1 and notice that in this case each agent always submits its c largest items. Indeed, submitting voluntarily smaller items the attained total weight would be dominated by switching to the larger items. So, in the remainder of this section, without loss of generality, under Rule 1 we assume $c = n$.

Theorem 1 *Problem* $CSS(R1, single)$ *has at most* $c + 1$ *efficient solutions. These can be computed in polynomial time.*

PROOF. Suppose that a feasible solution exists such that agent A wins k rounds and therefore B wins the remaining $c - k$ rounds. W.l.o.g. we assume $a_1 > b_1$ (otherwise we can exchange the roles of A and B). Since a_1 can win against any item of B, we always have $k \geq 1$ if A is rational. Otherwise, if A "voluntarily" decides to lose all rounds by submitting at each round an item smaller than the corresponding B item, k may also be equal to 0.

By a simple pair interchange argument it is easy to see that in this case the solution where A (B resp.) wins with its heaviest k ($c - k$ resp.) items is also feasible. Clearly, it is also nondominated. If $k \geq 1$, the structure of the resulting solution is such that item a_i is matched to b_{c-k+i} for $i = 1, \ldots, k$, and to b_{i-k} for $i = k + 1, \ldots, c$. Otherwise, for $k = 0$ we have an additional efficient solution with $w^A = 0$ and $w^B = \sum_{i=1}^{c} b_i$.

Computing the solution values for all $c + 1$ efficient solutions can be done in linear time by one scan through the list of items after sorting the items. Giving also the corresponding pairs of items explicitly as output would require $O(n^2)$ time.

A straightforward consequence of the above result is that, finding the global optimum, i.e. the solution M^* for which $w^A(M^*) + w^B(M^*)$ is maximum, can be done in a simple way in polynomial time also without solving a matching problem, since M^* is a Pareto efficient solution.

Even in the reusable items case, under Rule 1, finding the Pareto efficient solutions is an easy task. In particular, the following results can be proved similarly as above.

Theorem 2 *Problem* $CSS(R1, reuse)$ *has at most* $c + 1$ *efficient solutions. These can be computed in polynomial time.*

Clearly, it is easy to observe that, also in this case, a global optimum is attained when the best c items in $\{a_1, \ldots, a_n, b_1, \ldots, b_n\}$ win.

Turning to Rule 2, we observe that it is not a trivial task to select the "best"

c items to submit among the n available. Picking only large items may results in too many losses, a restriction to the smallest items increases the chances to win many rounds but the gain from these victories may be quite small. Hence, we first restrict our attention to the case $c = n$. In this case we can show the following results for the case with single-use items.

Theorem 3 *For $c = n$ Problem $CSS(R2, single)$ may have an exponential number of efficient solutions.*

Consider the following example for some small $\varepsilon > 0$.

Example 4

agent A		*agent B*		
a_1	0	b_1	ε	
a_i	2^{i-2}	b_i	$a_i + \varepsilon$	$i = 2, \ldots, c$

For any set $S \subseteq \{2, \ldots, c-1\}$ consider the solution M where an item a_i wins and, at the same time, b_i loses if and only if $i \in S \cup \{1\}$. It is always possible to construct such an M: clearly, a_1 always wins; then, if $i \in S$, a_i is matched to b_i, else ($i \notin S$) a_i is matched to some b_j, with $j < i$, $j \notin S$. In M, a_1 is matched to b_1 or b_c and a_c is always losing.

Neglecting the ε values, the sum of the weights $w^A(M)+w^B(M)$ is constant and equals $2^{c-1} - 1$. Therefore, for any possible choice of S, it is possible to obtain an efficient solution since the resulting values of $w^A(M)$ are all different. Thus, there are exponentially many efficient solutions.

We now show that, in the single-use items case under Rule 2, the problem of deciding whether a certain given objective for each agent can be achieved is $\mathcal{NP}$-complete by reduction from Partition. This implies that finding efficient solutions under Rule 2 is in general an $\mathcal{NP}$-hard task. To this purpose, we consider the following decision problem.

Recognition $CSS(R2, single)$:
Instance: positive integers a_i and b_i, $i = 1, \ldots, n$; two positive values W^A and W^B.
Question: Is there a solution (matching) M of $CSS(R2, single)$ such that $w^A(M) \geq W^A$ and $w^B(M) \geq W^B$?

Theorem 5 Recognition $CSS(R2, single)$ *is $\mathcal{NP}$-complete.*

PROOF. Consider an instance I of Partition with integer valued items $\{v_1, v_2, \ldots, v_n\}$ and $\sum_{i=1}^{n} v_i = V$. Assume $v_1 < v_2 < \ldots < v_n$.

For $T > v_n$ and $\varepsilon < 1/(n+1)$, build an instance I' of RECOGNITION $CSS(R2, single)$ as follows:

$$
\begin{array}{ll}
 & a_0 = T, \quad b_0 = \varepsilon, \\
(I') & a_i = v_i, \quad b_i = v_i + \varepsilon, \quad i = 1, \ldots, n, \\
 & W^A = W^B = \frac{V}{2}.
\end{array}
$$

Let I be a YES-instance of PARTITION. Then it is easy to build a solution M of I' such that $w^A(M) = V/2$ and $w^B(M) \geq V/2$. Let $S \subseteq \{1, 2, \ldots, n\}$ be such that $\sum_{i \in S} v_i = V/2$ and $\bar{S} = \{0, 1, \ldots, n\} \setminus S$. The required solution M is given as follows: Each item i of A with $i \in S$ is matched to the corresponding item i of B, while each item j of B with $j \in \bar{S}$ is matched to

$$\arg\min\{a_h \mid h \in \bar{S}, a_h > b_j\}.$$

Clearly, the two agents total weights are: $w^A(M) = \sum_{i \in S} a_i = V/2$ and $w^B(M) = \sum_{i \in \bar{S}} b_i \geq V/2$.

Now, let I' be a YES-instance of RECOGNITION $CSS(R2, single)$ and M be a solution of I' such that $\bar{S} \subseteq \{0, 1, 2, \ldots, n\}$ is the set of "winning" items of agent B. Note that $0 \in \bar{S}$, since item 0 of B always wins. So,

$$w^B(M) = \sum_{i \in \bar{S}} b_i = \sum_{i \in \bar{S}} a_i + \varepsilon \geq V/2$$

and $w^A(M) \geq V/2$. Note that, since each winning item i of A is matched to a losing item of B with larger weight, the total weight L^B of the "losing" items of agent B is such that $L^B = \sum_{i \in S} b_i \geq w^A(M) \geq V/2$. Since, $L^B + w^B(M) = V + (n+1)\varepsilon$, the v_i are all integers and $\varepsilon < 1/(n+1)$, then $\sum_{i \in S} v_i = \sum_{i \in \bar{S}} v_i = V/2$.

Observe that the above proof shows that the problem is $\mathcal{NP}$-hard even if $c = n$, i.e. the items to be submitted do not have to be selected from a larger ground set.

Although finding efficient solutions for $CSS(R2, single)$ is $\mathcal{NP}$-hard, the global optimum M^* can be found in polynomial time using matching techniques [4] as already mentioned in Section 1.2. However, it is easy to see that for any value of $c \leq n$, we may compute a global optimal solution in a straightforward way by setting $M^* = \{(a_1, b_1), (a_2, b_2), \ldots, (a_c, b_c)\}$. Indeed, assuming $a_1 > b_1$, A will lose with item a_1 in any case and B can score its largest possible gain. This argument can be repeated over all rounds.

Differently from above, in case of reusable items under Rule 2 finding Pareto efficient solutions becomes an easy task since the choice of the losing items has no consequence.

Theorem 6 *Problem $CSS(R2, reuse)$ has at most $c+1$ efficient solutions in general and at most c efficient solutions for $c = n$.*

PROOF. Assume there is a feasible solutions in which A wins exactly k rounds. Then, the following is the only (in terms of objective values) corresponding efficient solution. Without loss of generality, assume $b_1 > a_1$, let b_j be the largest among the B items such that $b_j < a_1$. Then, all B items $b_j, b_{j+1}, \ldots b_{j+c-k-1}$ are matched to a_1 (that loses). All items $a_1, a_2, \ldots a_k$ are matched to b_1.

For $c = n$ the case $k = 0$ is impossible since A will for sure win the round where B submits b_1.

2.2 Best-Worst Ratio

In this section we give an estimate on the ratio between the global optimum, i.e. the best efficient solution, and the value of *any* efficient solution. This concept is somehow connected to the notion of the *price of anarchy*, studied in algorithmic game theory, which is usually defined as the worst possible ratio between the value of the global (social) optimum and the value of a solution derived by a selfish optimization.

Here, we adopt a slightly different notion, called *best-worst ratio* defined as

$$\max_{M \in \mathcal{E}} \left\{ \frac{w^*}{w^A(M) + w^B(M)} \right\},$$

where $\mathcal{E}$ is the set of efficient solutions and w^* is the global optimum value.

In the following, we show that this ratio can be arbitrarily high for Problems $CSS(R2, single)$, $CSS(R1, reuse)$, and $CSS(R2, reuse)$ with $c < n$, while for Problems $CSS(R1, single)$ and $CSS(R2, reuse)$ with $c = n$ this ratio is limited.

Theorem 7 *The best-worst ratio of $CSS(R1, single)$ is 2.*

PROOF. Let $X = \{a_1, a_2, \ldots, a_c\} \cup \{b_1, b_2, \ldots, b_c\}$ and number the items of X in non-increasing order of weights, $X = \{x_1, x_2, \ldots, x_{2c}\}$. Obviously, the global optimum value is bounded from above by the sum of the c largest items,

i.e. $w^* \leq \sum_{i=1}^{c} x_i$. On the other hand, even the value of the efficient solution $\bar{M}$ with the *worst* global value can be bounded from below. In fact, in $\bar{M}$, the set X can be partitioned into the two sets X^V and X^L of winning and losing items. Clearly $|X^V| = |X^L| = c$. Since, under Rule 1, each winning item is matched to an item with smaller weight, then $w(X^V) \geq w(X^L)$. Therefore $w^A(\bar{M}) + w^B(\bar{M}) = w(X^V) \geq w(X)/2$. Since $w(X) \geq w^*$, the statement follows.

To show that the bound of 2 can be reached, consider the following example for a large constant T.

Example 8

agent A		*agent B*	
a_1	T	b_1	$T-1$
a_2	2	b_2	1

Let $c = 2$. Agent A submits items $\langle a_1, a_2 \rangle$. There are only two efficient solutions: If B submits b_1 first and b_2 second thus losing both rounds, the global solution value is $T + 2$. If B exchanges the order of its submissions then A wins the first round and B the second yielding a global solution value of $2T-1$. Therefore, for T tending to infinity, the best-worst ratio tends to 2.

Theorem 9 *The best worst ratio of Problem $CSS(R2, reuse)$ when $n = c$ is c.*

PROOF. Assuming $n = c$ and $b_1 > a_1$, there can be only c efficient solutions, since A must win at least one round, namely when B plays b_1 (see Theorem 6). Let $j' := \min\{j \mid b_j < a_1, j = 1, \ldots, n\}$. Each of the c efficient solutions has a total weight $w_k := \sum_{i=1}^{k} a_i + \sum_{j=j'}^{j'+c-k-1} b_j$ for $k = 1, \ldots, c$. It follows from the sorting of the weights and the definition of j' that $w_k \geq a_1$ and $w_k \leq c\, a_1$ for all $k = 1, \ldots, c$. Hence, the best worst ratio can be at most c.

To show that the bound of Theorem 9 is tight consider Example 10.

Example 10

agent A		*agent B*		
a_1	T	b_1	$T+1$	
a_i	1	b_i	$T-1$	$i = 2, \ldots, c$

If B sticks to submitting b_1 it always looses which yields an efficient solution with total weight $\sum_{i=1}^{c} a_i = T + (c-1)$. If A sticks to submitting a_1 and

B submits $\langle b_2, \ldots b_c, b_1 \rangle$ *we get another efficient solution with total weight* $\sum_{j=2}^{c} b_j + a_1 = (c-1)(T-1) + T$. *Obviously, for* T *tending to infinity the ratio between the total weights of these two efficient solutions approaches* c.

Negative results can be derived by using simple examples, as illustrated below.

Example 11

agent A		*agent B*	
a_1	T	b_1	$T+1$
a_2	2	b_2	1

Consider the problem when Rule 2 is in force and there is only one round. If agent B submits b_1 *it will lose in any case. Therefore, any selfish strategy of B will submit* b_2 *gaining in any case* $w^B = 1$, *while agent A will lose and have* $w^A = 0$. *A global optimum would submit* a_1 *and* b_1 *yielding a total weight of* T.

Example 11 above shows that the following theorem holds.

Theorem 12 *The best-worst ratio of Problem* $CSS(R2, reuse)$ *if* $n > c$ *can be arbitrarily high.*

Data of Example 8 and Example 11 can be used to show that the following theorem holds for $c = n$ and $c < n$, respectively.

Theorem 13 *The best-worst ratio of Problem* $CSS(R2, single)$ *can be arbitrarily high.*

Consider now the following example.

Example 14

agent A		*agent B*	
a_1	T	b_1	2
a_2	1	b_2	2

Let $c = 2$, *there are only two efficient solutions. Agent A submits items* $\langle a_1, a_2 \rangle$ *or* $\langle a_2, a_2 \rangle$. *In the first case the global solution value is* $T+2$, *while in the second it is* 4. *Therefore, for* T *tending to infinity, the best-worst ratio is unbounded.*

Example 14 above shows that the following theorem holds.

Theorem 15 *The best-worst ratio of* $CSS(R1, reuse)$ *can be arbitrarily high.*

It is easy to show that when losing items cannot be reused, under both rules, no Nash equilibria exist, except for trivial instances where only one Pareto optimum exists. The following example presents an instance in which there are no Nash equilibria in case of single-use items.

Example 16

agent A		*agent B*	
a_1	10	b_1	12
a_2	5	b_2	6

There are $c = n = 2$ rounds. Table 2 gives the normal-form representations of the game under the two rules. Assume, under Rule 1, that A submits a_1 and a_2 in the first and second round, respectively. Then the best sequence for B is $\langle b_1, b_2 \rangle$. Clearly, when B submits the latter sequence, A changes its own, by swapping a_1 with a_2. So, A's strategy $\langle a_1, a_2 \rangle$ is not in a Nash equilibrium. An analogous reasoning applies for the only other possible sequence for A, $\langle a_2, a_1 \rangle$. So, under Rule 1, no Nash equilibrium exists. The same argument applies for Rule 2.

	$\langle b_1, b_2 \rangle$	$\langle b_2, b_1 \rangle$
$\langle a_1, a_2 \rangle$	0,18	10,12
$\langle a_2, a_1 \rangle$	10,12	0,18

	$\langle b_1, b_2 \rangle$	$\langle b_2, b_1 \rangle$
$\langle a_1, a_2 \rangle$	15,0	5,6
$\langle a_2, a_1 \rangle$	5,6	15,0

Table 2
Payoffs of agents A and B with data of Example 16 under Rule 1 and 2 and single-use items.

When dealing with the reusable items case under Rule 1, the results are different. When both agents follow a greedy strategy (i.e. in each round, submit the largest available item) they reach a global optimum and the c overall largest items win. This solution correspond to a Nash equilibrium, since there is no convenience for one agent to deviate from the greedy strategy when the other agent follows the same greedy strategy.

In conclusion, the following theorem holds.

Theorem 17 *Problem $CSS(R1, reuse)$ admits a Nash equilibrium.*

It is an open question to determine a Nash equilibrium, if it exists, for Problem $CSS(R2, reuse)$.

3 Strategies of Agents

In this section we address the problem of devising a *strategy*, that is an algorithm that suggests an agent which item to submit at each round, in order to maximize the agent's weight under different information scenarios.

When dealing with the case where each item could be submitted only once (single-use-items case), the definition of a strategy of an agent is quite simple. It could be represented by an ordered list of c different items, called *submission list*, or by an algorithm producing it. Clearly, any such list is a feasible strategy.

When items are reusable and thus may appear several times in a submission list, its feasibility depends on the submissions of the other agent, since items cannot be reused after winning a round. Thus, no static list of items can be given as a strategy. Instead, we choose to define a strategy by an *algorithm* which generates the list of items submitted in every round depending on the currently still available items.

Here we consider two different "knowledge scenarios" for an agent, say B. In the first one, the strategy of agent A is completely unknown and we look for an algorithm guaranteeing the maximum weight in the worst possible case for B (*maximin* strategy). We show that for Rule 2 non-trivial algorithms can be developed while under Rule 1 it only remains for B to submit its largest items.

In the second scenario, agent B gets to know the item submitted by agent A in every round *before* making its own move. We will try to answer the question how B should select its items under such an advantageous asymmetry of information. In this particular context, the strategy adopted by agent B is referred to as *best response* strategy. Somewhat surprisingly, it will turn out that, in the single-use items case, for Rule 2 ($CSS(R2, single)$) an optimal best response strategy can be found while for Rule 1 ($CSS(R1, single)$) the existence of such a strategy can be ruled out even if an algorithm with an *a priori* guarantee on the quality of the solution can be found. An analogous result can be proven for the reusable items case under Rule 1 ($CSS(R1, reuse)$).

3.1 Maximin Strategies

In order to formally introduce the concept of maximin strategy, we give the following definitions. Let $\mathcal{S}_A$ ($\mathcal{S}_B$) be the set of strategies for agent A (B). For every pair of strategies $S_A \in \mathcal{S}_A, S_B \in \mathcal{S}_B$, $M(S_A, S_B)$ denotes the resulting solution. Then, a *maximin strategy* $MMS_B \in \mathcal{S}_B$ of agent B is defined as

follows:

$$MMS_B := \arg\max_{S_B \in \mathcal{S}_B} \left(\min_{S_A \in \mathcal{S}_A} W^B(M(S_A, S_B)) \right)$$

The definition of MMS_A is analogous.

In the single-use items case, i.e. for Problems $CSS(R1, single)$ and $CSS(R2, single)$, it can be observed that once the set $S(B) \subseteq \{b_1, \ldots, b_n\}$ of c items submitted by agent B is fixed, in the worst case agent A may minimize the weight attained by B by suitably matching its items to those in $S(B)$ (cf. Section 2.1). Hence, it does not matter in which order B submits its item set $S(B)$, it may always end up with the worst possible solution under $S(B)$. Let $\hat{w}^B(S(B))$ be the total weight for B in such a solution. It remains for a maximin strategy of agent B to determine its set of submitted items in the best possible way, i.e.

$$S(B) = \arg\max\{\hat{w}^B(X) \mid X \subseteq \{b_1, \ldots, b_n\}, |X| = c\}.$$

If $n = c$, there is nothing to choose since $S(B) = \{b_1, \ldots, b_n\}$, and no strategy can avoid to obtain the worst possible solution for B. In this case any strategy is technically a maximin strategy.

If $n > c$, under Rule 1, the maximin strategy of B obviously consists of choosing its c largest items, i.e. $S(B) = \{b_1, \ldots, b_c\}$. Under Rule 2, the selection of $S(B)$ is more involved and can be done by the following algorithm MaximinSingle 2.

Algorithm 1 Algorithm for agent B performing a maximin strategy for Problem $CSS(R2, single)$.

MaximinSingle 2

1: $\hat{w}^B := 0$
2: **for** $i = 1, \ldots, n - c + 1$ **do**
3: $S(B) := \{b_i, b_{i+1}, \ldots, b_{i+c-1}\}$
4: $S(A) := \{a_\ell \mid a_\ell \leq b_i\} \cap \{a_{n-c+1}, \ldots, a_n\}$
5: find the smallest item $b_k \in S(B)$ such that every item $b_\ell \in \{b_i, \ldots, b_k\}$ can be matched to an item a_j in $S(A)$ with $a_j \leq b_\ell$
6: $\hat{w}^B := \max\{\hat{w}^B, \sum_{\ell=k+1}^{i+c-1} b_\ell\}$ $\{k+1$ is the largest winning item in $S(B)\}$
7: **end for**
8: output the set $S(B)$ that defines $\hat{w}^B$

Theorem 18 MaximinSingle 2 *is a maximin strategy for* $CSS(R2, single)$.

PROOF. At first we will show that for an arbitrary set $S(B)$ submitted by B, the result of agent A acting in a malicious way aiming at minimizing the total gain of B will always lead to a bipartitioning of $S(B)$ such that A wins against the largest k items and looses against the remaining smallest $c - k$ items of

B for some k in $0, 1, \ldots, c$. Assume otherwise: If there is an item $b_\ell < b_{k+1}$ such that A wins against b_ℓ, then A could win with the same item against b_{k-1} instead, thus diminishing the weight won by B. Hence, the winning items of B must be a consecutive interval of $S(B)$.

Secondly, we show that it is always better (or at least not worse) for B to select a subset of items consisting of a consecutive subsequence of items from $\{1, \ldots, n\}$. Indeed, following from the above bipartition argument B wins all items smaller than some b_k and thus should always submit the *largest* $c - k$ items with weight $\leq b_k$ to maximize its gain, i.e. a consecutive interval. Moreover, since B will lose all items with weight $\geq b_k$, B can just as well submit the smallest $k-1$ items among these, yielding all together a consecutive interval.

It remains to compute the best such interval. Algorithm MaximinSingle 2 does so by simply checking each of the $n-c+1$ possibilities. For each candidate set $S(B)$ the "breakpoint" b_k is determined in line 5. This can be done by simply going down through the ordered set of items of $S(B)$ and finding a smaller, i.e. winning, matching partner from $S(A)$ for each of them. Obviously, as soon as there is no such partner for b_{k+1}, no item of $S(B)$ with even smaller weight can lose against A.

When dealing with maximin strategies, the possibility of reusing losing items, differently from what observed in the single-use items case, gives an agent more possibilities to maximize its weight in the worst case. We first consider Rule 1 and provide the following simple greedy algorithm.

Algorithm 2 Algorithm for agent B performing a maximin strategy for problem $CSS(R1, reuse)$.

MaximinReuse 1

1: In each round B submits the largest available item.

Theorem 19 MaximinReuse 1 *is a maximin strategy for* $CSS(R1, reuse)$.

PROOF. Consider the union of the two item sets $\{a_1, \ldots, a_n, b_1, \ldots, b_n\}$ and determine the subset containing the c largest items of this set. Assume that this subset consists of ℓ items of A and $c - \ell$ items of B.

Now consider the situation in which agent A applies the greedy strategy MaximinReuse 1 against an arbitrary strategy S_B of agent B. By the above definition there are at most $c - \ell$ items of B with weight larger than a_ℓ. Hence, no strategy of B can prevent A from winning items $a_1, \ldots, a_\ell$ by strategy MaximinReuse 1 during c rounds. It follows that for any strategy S_B agent B can win at most $c - \ell$ rounds against MaximinReuse 1. An upper bound on

the total solution weight of B playing against MaximinReuse 1 is therefore given by the largest items $\sum_{i=1}^{c-\ell} b_i$. This yields

$$\max_{S_B \in \mathcal{S}_B} \left(\min_{S_A \in \mathcal{S}_A} W^B(M(S_A, S_B)) \right) \leq \max_{S_B \in \mathcal{S}_B} W^B(M(S_A^*, S_B)) \leq \sum_{i=1}^{c-\ell} b_i \,,$$

where S_A^* is the strategy MaximinReuse 1 for A.

Now if B applies the greedy strategy MaximinReuse 1, it follows by a similar reasoning that no strategy of A can prevent B from winning items $b_1, \ldots, b_{c-\ell}$. Hence, for an arbitrary strategy of A the output of MaximinReuse 1 for B is not smaller than an upper bound on a maximin strategy which proves the theorem.

Turning to Rule 2, consider the following algorithm.

Algorithm 3 Algorithm for agent B performing a maximin strategy for Problem $CSS(R2, reuse)$.

MaximinReuse 2

1: **for** $i = 1, \ldots, n$ **do**
2: $\quad n^A(i) \leftarrow \min\{\, |\{a_j \mid a_j < b_i\}|\,,\, c\}$
3: $\quad n^B(i) \leftarrow \min\{c - n^A(i),\, n - i + 1\}$
4: **end for**
5: $i^* \leftarrow \arg\max_{i=1,\ldots,n} \sum_{k=i}^{i+n^B(i)-1} b_k$
6: In each round submit the largest available item with weight $\leq b_{i^*}$.

In line 2 we compute the number of items of A winning against b_i. Line 3 determines the number of items of B with weight at most b_i which agent B is guaranteed to win after A may have successfully submitted all $n^A(i)$ items beating b_i. The strategy indicated in line 6 starts with submitting b_{i^*} until it wins and proceeds with items $b_{i^*+1}, b_{i^*+2}, \ldots$.

Theorem 20
MaximinReuse 2 *is a maximin strategy for Problem* $CSS(R2, reuse)$.

PROOF. Consider a greedy strategy $G_{\min}$ for agent A consisting of submitting the *smallest* available item in every round in a game against an arbitrary strategy S_B of agent B. A will win some ℓ rounds out of c. By definition of $G_{\min}$ the winning items will be the ℓ smallest items of A. Agent B wins the remaining $c - \ell$ rounds. Let $b_{i'}$ be the largest of B's winning items under S^B. The total solution weight of B can not be better than $b_{i'}$ plus the weight of the $c - \ell - 1$ immediately following items. It follows from $G_{\min}$ that all items of A with $a_j < b_{i'}$ must have been submitted and won their rounds before $b_{i'}$ wins. In the notation of MaximinReuse 2 this implies $\ell \geq n^A(i')$.

Hence, the solution value of B against $G_{\min}$ can be bounded from above by a function of its largest winning item i' under S^B given by

$$W^B(M(G_{\min}, S_B)) \leq \bar{W}^B(i') := \sum_{k=i'}^{i'+c-\ell-1} b_k \leq \sum_{k=i'}^{i'+c-n^A(i')-1} b_k \,.$$

An upper bound over all strategies S^B is given by maximizing over the largest winning item i'. Altogether this yields

$$\max_{S_B \in \mathcal{S}_B} \left(\min_{S_A \in \mathcal{S}_A} W^B(M(S_A, S_B)) \right) \leq \max_{S_B \in \mathcal{S}_B} W^B(M(G_{\min}, S_B)) \leq \max_{i'=1,\ldots,n} \bar{W}^B(i') \,.$$

It is easy to see that line 5 of MaximinReuse 2 performs precisely this optimization task and i^* yields the best value for i'. Moreover, MaximinReuse 2 guarantees that B actually manages to win at least $\max_{i'=1,\ldots,n} \bar{W}^B(i')$, no matter which strategy A pursues. Hence, for an arbitrary strategy of A the output of MaximinReuse 2 is not smaller than an upper bound on a maximin strategy which proves the theorem.

It should be pointed out that MaximinReuse 2 follows a strictly worst-case point of view and may be improved to take advantage of strategies of A deviating from $G_{\min}$. Whenever an item b' submitted by B according to line 6 wins a round although there are still items available for A with weight $< b'$, (i.e. A missed a chance to win this round) the algorithm should start from the beginning and recompute item i^* for the remaining game. This may lead to an improved solution of B.

3.2 Best Response Strategies

We now consider the scenario where B only knows the submission of A in the current round and has to react immediately before both agents move on to the next round. In the following sections we show that for problems $CSS(R1, single)$, $CSS(R1, reuse)$, and $CSS(R2, reuse)$, there is no optimal strategy for B. Indeed we can show that no strategy of B can have a *competitive ratio* [2] compared to the best off-line strategy better than the golden ratio. Moreover, we show that, under Rule 1, a simple greedy-type algorithm has a worst-case tight competitive ratio of 2 (Sections 3.2.1 and 3.2.2). On the other hand, an algorithm yielding an optimal off-line solution for Problem $CSS(R2, single)$ exists (Section 3.2.3).

Note that, in the single-use items case, if B knows in advance the c items A submits (e.g., when $n = c$), it is easy to devise an optimal response algorithm. This task can be accomplished by solving an off-line maximum weight c-assignment problem (cf. Section 1.2).

3.2.1 Best Response for $CSS(R1, single)$

Assume B does not know in advance the c items A will submit (i.e., $n > c$ and A not rational). The following lower bound holds for any best response algorithm for $CSS(R1, single)$.

Theorem 21 *No on-line strategy of agent B against an arbitrary strategy of agent A can have a competitive ratio smaller than $\frac{\sqrt{5}+1}{2} = 1.618034\ldots$ for problem $CSS(R1, single)$.*

The example below proves Theorem 21.

Example 22

agent A		*agent B*	
a_1	2	b_1	$1+\varepsilon$
a_2	1	b_2	y
a_3	2ε	b_3	ε

There are $c = 2$ rounds and parameter y is chosen such that $2\varepsilon < y < 1$. We will consider two (suboptimal) strategies of agent A: In both cases A starts by submitting a_3. In strategy S_1 it is followed by a_1, in strategy S_2 by a_2. Given the first item a_3 there are only two ways for agent B to react:

In Case 1, B submits b_2 thus winning this round. Against strategy S_1, B loses the second round and the worst-case total weight for Case 1 is $W_1^B = y$.

In Case 2, B submits b_1 and wins again. But against strategy S_2, B loses again the second round and in the worst-case stops with a total weight $W_2^B = 1+\varepsilon$.

However, an optimal off-line strategy of B against S_1 would be $\langle b_1, b_2\rangle$ gaining a weight of $W_1^* = 1+\varepsilon$, while against S_2 the submission of $\langle b_2, b_1\rangle$ would yield $W_2^* = y+1+\varepsilon$. Altogether we get a lower bound for the worst-case competitive ratio of

$$\max_{2\varepsilon<y<1} \min_{i=1,2} \left\{ \frac{W_i^*}{W_i^B} \right\} = \max_{2\varepsilon<y<1} \min \left\{ \frac{1+\varepsilon}{y}, \frac{y+1+\varepsilon}{1+\varepsilon} \right\}.$$

Clearly, this maximum is attained if the expressions of the minimum are equal. An elementary calculation yields $y = (1+\varepsilon)\frac{\sqrt{5}-1}{2}$. Plugging in this value of y in the above equation yields $\frac{\sqrt{5}+1}{2}$ as a lower bound for the competitive ratio and proves Theorem 21.

In the above analysis B submits its $c = 2$ largest items; however it is not hard to see that the same result is attained when B may decide to submit b_3.

The following greedy-type algorithm tries to win against every item submitted by A with the largest item still available. If this fails and a loss cannot be avoided in the current round the smallest remaining item is submitted.

Algorithm 4 Algorithm for agent B responding to the submissions of agent A for Problem $CSS(R1, single)$.

GREEDY RESPONSE 1

$f \leftarrow 1$ {index of largest available item}
$\ell \leftarrow c$ {index of smallest available item}
repeat
  A submits item a'
  if $b_f > a'$ **then**
    B submits b_f and wins
    $f \leftarrow f + 1$
  else
    B submits b_ℓ and loses
    $\ell \leftarrow \ell - 1$
  end if
until $f < \ell$ {all c rounds finished}

Theorem 23 *Algorithm* GREEDY RESPONSE 1 *has a tight competitive ratio of* 2 *for Problem* $CSS(R1, single)$.

PROOF. Let S^A be the complete set of items submitted by A. At the end of the execution of GREEDY RESPONSE 1 agent B has added its k largest items into the solution and reached a total weight of W^B ($k = f - 1$ at the end of the algorithm).

It is easy to see that an optimal off-line solution against the same set S^A will always consist of the largest k^* items with $k^* \geq k$. In this optimal strategy the items in set $D^B = \{k+1, \ldots, k^*\}$ of B all win against some items in S^A. This means that at least the smallest items in S^A must be potential losers against D^B. Let j be the index of the $c - (k^* - k - 1)$ largest item in S^A. Then we have $b_{k+1} > a_j$.

Considering the item $b_f = b_\ell$ submitted by B in the last round of the game there are two cases to distinguish.

Case 1: b_f loses. Since $b_f = b_{k+1}$ would win against any of the $k^* - k$ smallest items in S^A, these must have been submitted by A before. Moreover, at the rounds where these items were submitted, item b_{k+1} was still available and would yield a win for B. By definition of GREEDY RESPONSE 1 some other item of B must have won these rounds. Therefore, the total number of wins for B must be at least $k^* - k$, i.e. $k \geq k^* - k$, which proves the statement of the theorem.

Case 2: b_f wins. In all rounds, in which A submitted one of the $k^* - k$ smallest items in S^A, item b_f was still available and would have won against these items. As above, by the definition of GREEDY RESPONSE 1 some other item of B must have won these rounds and again $k \geq k^* - k$.

To show that the bound of 2 is tight consider the simple Example 24.

Example 24

agent A		*agent B*	
a_1	$1+\varepsilon$	b_1	$1+2\varepsilon$
a_2	$1-\varepsilon$	b_2	1

There are $c = 2$ rounds. Agent A submits a_2 in the first round. GREEDY RESPONSE 1 *reacts with b_1 and loses the second round (a_1 against b_2) gaining a total weight of $1 + 2\varepsilon$.*

An optimal off-line strategy of B would react with b_2 in the first round thus winning both rounds and gaining $2 + 2\varepsilon$.

3.2.2 Best Response for $CSS(R1, reuse)$

When items are reusable under Rule 1, we have similar results to the single-use items case described in Section 3.2.1. In particular, Algorithm GREEDY RESPONSE 1 guarantees the same competitive ratio.

Theorem 25 *Algorithm* GREEDY RESPONSE 1 *has a tight competitive ratio of 2 for Problem $CSS(R1, reuse)$.*

Observe that, since it does not matter which item is used to lose, the algorithm can be simplified: at each round B tries to win with the largest available item. The proof of Theorem 25 is omitted since that of Theorem 23 applies. Moreover, with the same arguments we can show that the golden ratio lower bound of Theorem 21 holds for Problem $CSS(R1, reuse)$.

3.2.3 Best Response for $CSS(R2, single)$

As in Rule 1, the following greedy-type algorithm for Rule 2 tries to win against every item submitted by A with the largest possible winning item. If no such item exists, a losing item is determined which can not worsen the remainder of the solution for B. To avoid a tedious special treatment of ties we will assume in this subsection that all $2n$ items have different weights.

Algorithm 5 Algorithm for agent B responding to the submissions of agent A for Problem $CSS(R2, single)$.

GREEDY RESPONSE 2

$R^A \leftarrow \{a_1, \ldots, a_n\}$ {set of remaining items for A}
$R^B \leftarrow \{b_1, \ldots, b_n\}$ {set of remaining items for B}
repeat
  A submits item a', remove a' from R^A
  $b_{\min} \leftarrow \min\{b_i \mid b_i \in R^B\}$
  if $a' > b_{\min}$ **then**
    $b_{win} \leftarrow \max\{b_i \mid b_i < a', b_i \in R^B\}$
    B submits b_{win} and wins, remove b_{win} from R^B
  else
    $A_j \leftarrow \{a_i \mid a_i \geq b_j, a_i \in R^A\},\ j = 1, \ldots, |R^B|$
    $B_j \leftarrow \{b_i \mid b_i \geq b_j, b_i \in R^B\},\ j = 1, \ldots, |R^B|$
    determine the minimum index $\ell \geq 1$ such that $|A_\ell| < |B_\ell|$
    B submits b_ℓ and loses, remove b_ℓ from R^B
  end if
until all c rounds finished

Theorem 26 *Algorithm* GREEDY RESPONSE 2 *yields the optimal off-line solution of Problem* $CSS(R2, single)$.

PROOF. We will consider round 1, where agent A submits a', and show that also an optimal algorithm OPT can not do better than GREEDY RESPONSE 2 and that the later does not diminish the range of options for an optimal strategy in the subsequent rounds. Repeating this argument over all rounds yields the statement of the theorem.

Case 1: B wins. What are the alternatives for OPT? If OPT submits an item $b_j < b_{win}$ it earns a smaller weight in round 1. In contrary to GREEDY RESPONSE RULE 2 it can use b_{win} to win in one of the subsequent rounds. However, in this case GREEDY RESPONSE 2 could submit b_j in this future round and win as well since $b_j < b_{win}$. Obviously, the total weight of these two rounds would be the same for both algorithms.

If OPT chooses to submit an item $b_k \geq a'$ and loose "voluntarily", it can again use b_{win} to win in a later round. GREEDY RESPONSE 2 might be forced to loose in that round, but again the two algorithms end up with the same total weight for these two rounds.

Case 2: B looses. First, let us note that in the special case $b_1 > a_1$ (considering only items in R^A and R^B) GREEDY RESPONSE 2 will submit $b_\ell = b_1$. Since b_1 cannot win in any round, it clearly cannot violate optimality to submit b_1 in a round which B will lose in any case.

Also for $b_1 < a_1$ index ℓ is well defined. Note that $b_j \in B_j,\ \forall\ j$. For $j = 1$ the defining inequality in line 12 is clearly violated since $A_1 \supseteq \{a_1\}$ and $B_1 = \{b_1\}$. Since $a' < b_{\min}$ and $|R^A| = |R^B|$, an index ℓ satisfying $|A_\ell| < |B_\ell|$ will be determined for $b_{\min}$ at the latest.

Note that in the ordering of the items in $R^A \cup R^B$ there must occur an item $b_{\ell-1} \in R^B$ directly before b_ℓ. Otherwise, since there was $|A_{\ell-1}| \geq |B_{\ell-1}|$, one (or more) items of R^A plugged in between $b_{\ell-1}$ and b_ℓ would not lead to the required change in the balance of cardinalities. This means that $A_{\ell-1} = A_\ell$.

Now consider that $B_{\ell-1}$ can be matched to win against $A_{\ell-1}$. This can be seen by a simple scan through the items in $R^A \cup R^B$ in decreasing order, starting with a_1 being matched to b_1. A failure of such a scan would immediately yield a new smaller value of ℓ. In fact, the existence of such a matching follows from Hall's marriage theorem. Clearly, also every subset of items of $A_{\ell-1}$ $(= A_\ell)$ can be matched to be beaten by the corresponding number of items of $B_{\ell-1}$ (e.g. by taking the smallest items in $B_{\ell-1}$).

On the other hand, it follows from the definition of $|A_\ell|$ and $|B_\ell|$ that there are more items in R^B with weight larger or equal b_ℓ than in R^A. Therefore, it is impossible for any strategy that *all* items in B_ℓ will win against some items in R^A, the items in A_ℓ being the only candidates for losers.

Since it is impossible for agent B to beat all items in A_ℓ, but every strict subset of A_ℓ can be beaten also by items in $B_{\ell-1}$, the removal of b_ℓ does not change the options of Opt in any subsequent round.

3.2.4 Best Response for $CSS(R2, reuse)$

In this section, we show that for Problem $CSS(R2, reuse)$, no best response strategy of B can have a competitive ratio (compared to the best off-line strategy) better than the golden ratio.

Theorem 27 *No on-line strategy of agent B against an arbitrary strategy of agent A can have a competitive ratio smaller than $\frac{\sqrt{5}+1}{2} = 1.618034\ldots$ for problem $CSS(R2, reuse)$.*

The above result can be proved by considering the following example.

Example 28

agent A		*agent B*	
a_1	2	b_1	$1+\varepsilon$
a_2	$1-\varepsilon$	b_2	y
a_3	ε	b_3	$2\,\varepsilon$

There are $c = 3$ rounds. Assume A can use one of the two following strategies: S_1 considers the items in the following order $\langle a_2, a_3, a_1\rangle$ and submits each item until it wins. S_2 submits the items a_2, a_1 and a_3 in rounds 1, 2 and 3, independently from the outcome of these rounds. (Since the arguments are very similar to those presented in Example 22, hereafter we omit the treatment of all the cases which are dominated from a lower bound computation point of view.)

In the first round B can answer either with $b_2 < a_2$, thus winning this round (Case 1), or it can answer with b_1, losing the round (Case 2). In Case 1, A will submit a_2 again in the second round, B can answer with b_3 and win (Case 1.1). Then A would submit a_2 in the third round and finally win a_2. So, in Case 1.1, B's solution has a total weight $W_1^B = b_2 + b_3$, while an optimal solution would have weight $W_1^* = b_1$. If B answers with b_1 in the second round (Case 1.2), the resulting outcome is dominated by Case 1.1 from a lower bound point of view.

In Case 2, A will submit a_1 in the second round according to S_2 and B's best answer is b_1. In the third round A will submit a_3 and B will answer with b_2. Summing up Case 2, B's solution has weight $W_2^B = b_1$, while an optimal solution would have weight $W_2^* = b_1 + b_2$.

Altogether we get a lower bound for the worst-case competitive ratio of

$$\max_{2\varepsilon<y<1-\varepsilon} \min_{i=1,2} \left\{ \frac{W_i^*}{W_i^B} \right\} = \max_{2\varepsilon<y<1-\varepsilon} \min \left\{ \frac{1+\varepsilon}{y+2\varepsilon},\ \frac{y+1+\varepsilon}{1+\varepsilon} \right\}.$$

Similarly to Example 22, for ε tending to 0, the maximum is obtained for $y = \frac{\sqrt{5}-1}{2}$. Plugging in this value of y in the above equation yields $\frac{\sqrt{5}+1}{2}$ as a lower bound for the competitive ratio and thus proves Theorem 27.

4 Conclusions

In this paper we addressed a multi-agent optimization problem where two agents compete to add items in a solution set. We consider four different versions of the problem and, for all versions, we give results concerning the com-

putational complexity, the structure of efficient solutions, and some strategies for optimizing the objective of a single agent against the other.

It is still an open question to determine a Nash Equilibrium, if it exists, for problem $CSS(R2, reuse)$. It seems that the difficulty lies in the definition of strategy in this particular scenario. Another interesting question concerns the design of a best response algorithm again for problem $CSS(R2, reuse)$ with a limited competitive ratio, since an optimal response algorithm does not exist.

This work puts several directions forward for future research. One is the design of algorithms to efficiently enumerate all Pareto-optimal solutions for problem $CSS(R2, single)$. Another is to extend our results to further relevant rules controlling how an agent's item is added to the solution set.

Furthermore, it would be interesting to investigate the problem with incomplete information, i.e. where both agents ignore the weights of the items of the other agent.

References

[1] G. Aggarwal, J. D. Hartline. Knapsack auctions, Proceedings of the 17th annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1083–1092, 2006.

[2] A. Borodin, R. El-Yaniv. Online Computation and Competitive Analysis, Cambridge University Press, 1998.

[3] L. Brotcorne, S. Hanafi, R. Mansi. A dynamic programming algorithm for the bilevel knapsack problem, Operations Research Letters **37**, pp. 215–218, 2009.

[4] M. Dell'Amico, S. Martello. The k-cardinality assignment problem, Discrete Applied Mathematics **76**, pp. 103–121, 1997.

[5] I. Fleming. James Bond Omnibus, Titan Books Ltd., 2009.

[6] M. Fujimoto, T. Yamada. An exact algorithm for the knapsack sharing problem with common items, European Journal of Operational Research **171**, pp. 693–707, 2006.

[7] M. Hifi, H. M'Hallab, S. Sadfi. An exact algorithm for the knapsack sharing problem, Computers and Operations Research **32**, pp. 1311–1324, 2005.

[8] J. Kahn, J. C. Lagarias, H. S. Witsenhausen. Single-suit two-person card play, International Journal of Game Theory **16**, pp. 291–320, 1987.

[9] N. Katoh, J. Koyanagi, M. Ohnishi and T. Ibaraki. Optimal strategies for some team games, Discrete Applied Mathematics **35**, pp. 275–291, 1992.

[10] V. Liberatore. Scheduling jobs before shut down, Lecture Notes in Computer Science **1851**, Algorithm Theory - SWAT 2000, pp. 461–466, Springer, 2000.

[11] K.H. Schlag, A. Sela. You play (an action) only once, Economic Letters **59**, pp. 299–303, 1998.

[12] J. Wästlund. A solution of two-person single-suit whist, The Electronic Journal of Combinatorics **12**, R43, 2005.