NEW ALGORITHMS FOR SOLVING SINGLE-ITEM REVERSE AUCTION

Dang Thanh Tung(1), Baltazár Frankovič(1), Con Sheahan(2), Ivana Budinská(1)

(1) Institute of informatics, Slovak Academy of Sciences, Dubravska 9, Bratislava 84507, Slovakia
(2) Dept. Of Manufacturing & Operations Eng., University of Limerick, Limerick, Ireland.
Email: utrrtung@savba.sk

Abstract: This paper deals with the “reverse auction” problem with an assumption sellers are willing to offer quantity discounts to the buyer. The objective of the buyer is to find such an allocation of quantities that one should buy from each seller, in order to pay as little as possible for the given amount of any product. Two algorithms are presented to solve the introduced problem. The first algorithm is able to find the global optimal solution, although only for some classes of cases. The second algorithm has pseudo-linear complexity and it achieves sub-optimal solutions within predictable range of the optimal one. *Copyright © 2005 IFAC*

Keywords: Agents, heuristic search, optimization, auction.

1. INTRODUCTION

This paper deals with the problem called “reverse auction”, in which one buyer and many sellers participate in trading. The objective of the buyer is to buy a certain amount of products with as low cost as possible from the candidate sellers. On the other hand, sellers are competitive with each other and therefore they might try to offer various kinds of advantages. One of the most used methods is to offer the discount unit price for buying larger amount of products. The offer that each seller sends to the buyer usually includes these discount prices and corresponding conditions – i.e. the minimal amount that the buyer should buy, in order to get this discount price. From a number of such offers, the buyer makes a choice and distributes the amount of products that he/she wants to buy from each seller so that the total payment would be minimal.
The buyer decides for his/her own benefits, so the amounts of products or the unit prices that he/she is willing to pay do not have to be equal for all sellers. The problem solving could be drastically simplified if the buyer does not discriminate the sellers and agrees to buy the same amount or to pay the same unit price to each seller.

Due to the fact that the offer of each seller is usually a discrete function and the amount of products the buyer can buy is an integer value, solving the introduced optimization problem is usually based on using heuristic search methods. In this paper, two algorithms will be presented to solve the above problems with some specified simplifications.

2. LITERATURE OVERVIEW

The reverse auction problem has been dealt in many papers, e.g. (Gonen & Lehmann, 2000), (Sandholm, et. al, 2002), (Sandholm and Suri, 2002). Many real cases are realized, for example, in http://reverse.interauct.com.sg/. However, (Sandholm et al. 2002) showed that in certain variants even finding a feasible solution is *NP*-complete. For a special case, when the price function is linear, (Sandholm and Suri, 2002) proposed a polynomial algorithm to solve. [Dang & Jennings, 2002] is closest to this work, in which an algorithm with O(n^2) was presented to find sub-optimal solution within predictable range. The coefficient of prediction seems too high; essentially it increases exponentially when the problem solving is multi-item trading. For that reason, the focus is to propose a new algorithm that is able to find a solution in polynomial time but with better estimated coefficient (closer to the really optimal solution). Before starting the discussion, some assumptions should be presented, in order to simplify the solving problem.

Assumption 1: The unit price does not increase when the bought quantity increases.

Assumption 2: The total payment does not decrease when the bought quantity increases.

These assumptions are based on the reality, because sellers usually tend to decrease a unit price when someone buys larger amount of product. In addition, sellers do not want to loose, thus, the total payment should not decrease, even though the bought amount increases.

3. SINGLE ITEM FOR TRADING

The problem solving could be described as follows: given *n* sellers $\{S_i\}_{i=1,..,n}$ and one buyer B; the objective is that the buyer is to buy Q (Q∈N) units of one product with as low cost as possible. Each offer provided by seller S_i includes *k* pairs $\{a^i_j, p^i_j\}_{j=1,..,k}$, which express the price per one unit and the minimal amount to be bought in order to apply this price. For all offers the following condition is valid:

$$\forall i: a^i_1=0 \text{ and } a^i_k \le Q. \qquad (1)$$

Let us denote $p_i(q)$ as a unit price that seller S_i offers if the bought quantity is equal to q; then, based on Assumption 1,

$$\forall i, \text{ if } q'>q'' \text{ then } p_i(q') \le p_i(q''). \qquad (2)$$

Applying this condition to the given case we get:

$$\forall i,j: p_i(a^i_j)>p_i(a^i_{j+1}). \qquad (3)$$

Additionally, on the basis of Assumption 2, the following condition is derived:

$$\forall i, \text{ if } q'>q'' \text{ then } q'p_i(q') \ge q''p_i(q''). \qquad (4)$$

Let $\{q_1,\ldots,q_n\}$ be a vector of quantities that the buyer buys from each seller; then

$$\sum_{i=1}^{n} q_i = Q. \qquad (5)$$

The total payment F is equal to

$$F=\sum_{i=1}^{n} q_i p_i(q_i). \qquad (6)$$

The main goal is to find such a vector of quantities which minimizes the F value from (6) subject to condition (5).

4. REDUCTION OF SOLUTION SPACE

First, some proofs are presented with the purpose to explain the principle of the later-proposed algorithms.

Let us consider a case with two sellers $\{S_1, S_2\}$. q_1 and q_2 denote the quantities bought from each seller,respectively; $q_1 + q_2 = Q$ and $q_1 \in [\alpha_1, \alpha_2]$; $q_2 \in [\beta_1, \beta_2]$, where

$$\alpha_2 \ge Q \text{ and } \beta_2 \ge Q. \qquad (7)$$

That means both sellers are able to provide the required amount. Without loss of generality we assume that $p_1(Q) \ge p_2(Q)$. From (2) it is easy to get

$$\forall q_1, q_2: p_1(q_1) \ge p_2(Q) \text{ and } p_2(q_2) \ge p_2(Q) \qquad (8)$$

This leads to the following result:

$$F = q_1p_1(q_1)+q_2p_2(q_2) \ge Qp_2(Q). \qquad (9)$$

Equality occurs when $q_1=0$ and $q_2=Q$. That means the total payment is minimal when $q_1=0$ and $q_2=Q$. Similarly, with case $p_1(Q)< p_2(Q)$. Based on this proof the following lemma is derived for general case with many sellers:

Lemma 1: Given any demand Q∈N. If all sellers are able to provide this demand, then the payment is minimal when only one seller is selected to buy. □

Consequence 1: Denote $a_{min}= \min\{a^i_k\}_{i=1,..,n}$. The maximal number of sellers who contribute to the desired demand Q with minimal total payment is $[2Q/a_{min}]+1$.

Consequence 2: There is maximally one value q_i, which is lower than $a_{min}/2$.

Proof: follows from Lemma 1.

Consequences 1 and 2 are used to restrict the set of seller's configurations that the solver has to explore.

In case when each seller offers one unit price, the solution is achieved very quickly.

Theorem 1: Let us assume that the quantities creating the optimal solution $\{q_i\}_{i=1,..,n}$ satisfy the following conditions:

$$\forall i: q_i \in [l^i_{x_i}, h^i_{x_i}] \qquad (10)$$

where ∀i, $x_i \in [0,k]$, $l^i_{x_i} = a^i_j$ and $h^i_{x_i} = a^i_{j+1}-1$ are borders of any interval in which the unit price provided by S_i is constant. Then, the optimal solution could be achieved in O(*n**log*n*).

Proof: First, we deduce the value of the optimal solution which minimizes the F value. To simplify, we denote unit prices offered by sellers as $\{p_1\}_{i=1,..,n}$ (∀i, $p_i= p^i_{x_i}$). Let us sort out unit prices from lowest to the highest. Let us assume that after sorting

$$p_1 \le p_2 \le \ldots \le p_n \qquad (11)$$

Let us consider any solution $\{q_i\}_{i=1,..,n}$. Sequentially, let us perform the following operation: simultaneously increasing q_1 and decreasing q_n by 1 leads to another solution, but clearly with not higher F value. Repeat this process until $q_1 = h^1_{x_1}$ or $q_n = l^n_{x_n}$. If q_1 reaches its top border earlier than q_n reaches its low border, repeat the same operation with q_2 and q_n; in the opposite case, continue this process with q_1 and q_{n-1}, until one of them reaches its border, etc. Finally, this procedure converts to the solution $\{h^1_{x_1}, h^2_{x_2}, \ldots, q_r, \ldots, l^{n-1}_{x_{n-1}}, l^n_{x_n}\}$ where at most only one member q_r is not a border value. It is easy to prove that the final solution has the lowest F value among all possible configurations, in which the quantities $\{q_i\}_{i=1,..,n}$ satisfy the conditions presented in (10). The member q_r divides the final solution into two parts; the left part consists of (*r*-1) top boundary values of intervals in which the unit price is lower than or equal to p_r; and the right part consists of (*n-r*) low boundary values of intervals in which the unit price is higher than or equal to p_r. The value *r* is called breakpoint of the solution. In order to get the solution, it suffices to identify the value of breakpoint *r*.

Define the function

$$f(z)=\sum_{i=1}^{z} h^i_{x_i}+\sum_{i=z+1}^{n} l^i_{x_i} \text{ where } z\in[0,n]. \tag{12}$$

Because $q_r \in [l^r_{x_r}, h^r_{x_r}]$, it follows:

$$f(r-1)\le \sum_{i=1}^{r-1} h^i_{x_i} + q_r + \sum_{i=r+1}^{n} l^i_{x_i} = Q\le f(r) \tag{13}$$

(13) points out that to find the value *r*, which divides the final solution into two parts, it suffices to compare the values f(z) and f(z-1) with Q, until inequalities (13) are satisfied. Because f(z) is an increasing function in the interval [0,n], it is possible to find the value *r*, which satisfies inequalities (13) with ≅ O(log*n*). On the other hand, sorting unit prices $\{p_i\}_{i=1,..,n}$ requires O(*n**log*n*) operations. That fact leads to the conclusion that an optimal solution with known unit prices (or domain of quantities) requires O(*n**log*n*) operations. The theorem is proved.

Theorem 1 shows a uniform approach to achieve the optimal solution with O(*n**log*n*); however, on the assumption that each seller offers only one unit price. In order to achieve the optimal solution, all configurations of unit prices (or quantities domains satisfying (10)) must be explored. Although the number of possible configurations is very large, many of them could be omitted by using the following lemmas.

Lemma 2: Let us assume that quantities $\{q_i\}_{i=1,..,n}$ satisfy the condition in (10). A solution (not necessarily the optimal one) exists if and only if:

$$\sum_{i=1}^{n} l^i_{x_i} \le Q \le \sum_{i=1}^{n} h^i_{x_i} \tag{14}$$

Proof: =>) if any solution exists, then $\sum_{i=1}^{n} l^i_{x_i} \le \sum_{i=1}^{n} q_i = Q \le \sum_{i=1}^{n} h^i_{x_i}$.

<=) if the conditions in (14) are valid. Set ∀i: $q_i = l^i_{x_i}$ and define the variable $f=\sum_{i=1}^{n} q_i$. Sequentially do the cycle i=1, $q_i=q_i+1$, update the value *f*, and continue until $q_i=h^i_{x_i}$. Increase i=i+1 and repeat the above cycle, until i=n. It is clear that the value *f* increases by 1 after each operation, from starting $\sum_{i=1}^{n} l^i_{x_i}$ up to $\sum_{i=1}^{n} h^i_{x_i}$. As a result, there must be a case when *f*=Q, i.e. there must be $\{q_i\}_{i=1,..,n}$ such that $\sum_{i=1}^{n} q_i$ =Q. The lemma is proved.

To simplify, in following Lemmas 3 and 4 we assume that quantities $\{q_i\}_{i=1,..,n}$ satisfy the conditions in (10) and (14). In addition, $\{p_i\}_{i=1,..,n}$ denote the unit prices offered by sellers, i.e. ∀i, $p_i=p^i_{x_i}$; and the approach presented in the proof of Theorem 1 is used to construct a final solution. These lemmas show the needed conditions guaranteeing the optimality of the achieved solution.

Lemma 3: If the achieved solution is the optimal one, then one of the following conditions is valid:

1. $$\sum_{i=1}^{n} l^i_{x_i} = Q \tag{15}$$

or

$$\sum_{i=1}^{n} h^i_{x_i} = Q$$
$$\text{and } h^j_{x_j} = a^j_k \text{ where } p_j = \min\{p_i\}_{i=1,..,n} \tag{16}$$

2. $$\sum_{i=1}^{n} l^i_{x_i} \le Q \le \sum_{i=1}^{n} l^i_{x_i} - l^j_{x_j} + h^j_{x_j} \tag{17}$$

where $p_j=\min\{p_i\}_{i=1,..,n}$, or

$$\sum_{i=1}^{n} h^i_{x_i} \ge Q \ge \sum_{i=1}^{n} h^i_{x_i} - h^t_{x_t} + l^t_{x_t}$$
$$\text{and } h^j_{x_j} = a^j_k \tag{18}$$

where $p_t = \max\{p_i\}_{i=1,..,n}$, $p_j= \min\{p_i\}_{i=1,..,n}$,

3. $h^j_{x_j} = a^j_k$ (maximal possible amount that S_j can provide), where $p_j=\min\{p_i\}_{i=1,..,n}$.

Proof: The approach presented in the proof of Theorem 1 could lead to one of the following situations:

1. f(0)=Q or f(n) =Q, i.e. ∀i, $q_i = l^i_{x_i}$ or $q_i = h^i_{x_i}$. Consequently, either(15) or (16) will be valid.
2. f(0)<Q≤f(1), that means the solution consists of one quantity $h^j_{x_j} \ge q_j > l^j_{x_j}$ where $p_j=\min\{p_i\}_{i=1,..,n}$ and all low borders of the rest of quantities, i.e. $q_i|_{i\ne j} = l^i_{x_i}$. In this case, equation (17) will be valid.

Similarly it is provable that equation (18) will be valid in case when f(n-1)≤Q<f(n).

The proof of the second part of (16) and (18) is based on the same approach as presented below.

3. f(r-1)<Q≤f(r), where n-1≥r≥2. Since f(1)<Q, there must be at least one q_j which has top border value, i.e. $q_j = h^j_{x_j}$, where $p_j=\min\{p_i\}_{i=1,..,n}$ and another q_s which is larger than its low border value, i.e. $h^s_{x_s} \ge q_s > l^s_{x_s}$ (p_s could be the second lowest unit price following p_j). If $q_j<a^j_k$(the maximal possible amount provided by seller S_j), then q_j increases and simultaneously q_s decreases by 1, i.e. $q_j=q_j+1$ and $q_s=q_s-1$, the unit price p_s does not change (due to new $q_s \ge l^s_{x_s}$), but the unit price offered by seller S_j is now lower, i.e. $p_j' < p_j$. The payment for seller S_j increases by not more than p_j'; on the other hand, the payment for seller S_s decreases by p_s, where $p_s \ge p_j > p_j'$. As a result, the new solution will have lower total payment than the original one. That contradicts the given assumption which says the achieved solution is the optimal one. The lemma is proved.

Lemma 4: ∀i∈[1,n], x_i∈[1,k], we then

$$\Delta^i_{x_i} = l^i_{x_i}\, p^i_{x_i} - (l^i_{x_i}-1)\, p^i_{x_i-1} \tag{19}$$

If the achieved solution is the optimal one, then

$$p_{\min} \ge \max\{\Delta^i_{x_i}\}_{i=1,..,n} \tag{20}$$

where $p_{\min}$ is defined as the minimal unit price of all sellers, whose top border $h^i_{x_i}$ is not the maximal amount (or the unit price is not the lowest one) that this seller can offer:

$$p_{\min} = \min\{p_i \mid h^i_{x_i} < a^i_k\} \tag{21}$$

To prove this lemma, the following proposition is needed.

Proposition 1: Consider any seller S_i, the bought quantity is q_i corresponding to unit price $p(q_i)$. If the quantity increases by 1 unit, the payment for this seller does not increase more than $p(q_i+1)$.

Proof: The additional payment when the quantity increases by 1 unit is

$$\Delta=(q_i+1)p(q_i+1) - q_i p(q_i) \qquad (22)$$

From (2) and (4), $p(q_i+1)\leq p(q_i)$. As a result,

$$\Delta= p(q_i+1)+q_i[p(q_i+1) - p(q_i)] \leq p(q_i+1). \qquad (23)$$

The proposition is valid.

Proof of Lemma 4: Without loss of generality, assume that $p_1\leq p_2\leq\ldots\leq p_n$. The final solution has the form $\{h^1_{x_1}, h^2_{x_2}, \ldots, q_r, \ldots, l^{n-1}_{x_{n-1}}, l^n_{x_n}\}$, where there is maximally one q_r which is not a border value, i.e. $l^r_{x_r} < q_r < h^r_{x_r}$ and $\forall i\in[1,r-1]$: $q_i = h^i_{x_i}$; $\forall i\in[r+1,n]$: $q_i = l^i_{x_i}$. By applying the same approach as in the proof of Lemma 3, Point 3, it is possible to prove that:

1. if $q_r > l^r_{x_r}$, then $\forall i=1,..,r-1$: $h^i_{x_i} = a^i_k$;
2. if $q_r = l^r_{x_r}$, then $\forall i=1,..,r-2$: $h^i_{x_i} = a^i_k$;

If case 1 occurs, two situations have to be considered: (1.a) $h^r_{x_r} < a^r_k$, and (1.b) $h^r_{x_r} = a^r_k$.

(1.a): it is clear that $p_r=p_{min}$; then we will show that

$$\forall i=r+1,..,n\text{: } p_r\geq \Delta^i_{x_i} \qquad (24)$$

Let us take away one unit from any $q_i|_{i=r+1,..,n}$ and add one 1 unit to q_r. According to Proposition 1, the additional payment for seller S_r is not more than the unit price that S_r offers for quantity (q_r+1), so it is not more than p_r. On the other hand, the reduction of the payment for seller S_i is equal to $\Delta^i_{x_i}$. Due to the assumption that the achieved solution is the optimal one, the total payment for new configuration must not be lower than the original one. Consequently, $\forall i\in[r+1,n]$: $\Delta^i_{x_i} \leq p_r$, or (24) is valid. Applying Proposition 1 to a situation when $q_i = l^i_{x_i} -1$, the following result is achieved:

$$\forall i,\ \Delta^i_{x_i} \leq p^i_{x_i}. \qquad (25)$$

Since $p_1\leq p_2\leq \ldots\leq p_r$, we get:

$$\forall i=1,..,r\text{: } \Delta^i_{x_i} \leq p_r \qquad (26)$$

By combining (24) and (26), (20) is achieved.

(1.b): in this case $p_{min} = p_{r+1}$. By using the same approach as above (increasing q_{r+1} and decreasing any $q_i|_{i=r+2,..,n}$ by 1), (20) will be achieved.

Case 2 is similar to case 1.b, so the proof is omitted.

Lemmas 3 and 4 are useful for checking the optimality of the achieved solution, before the solving process is started. Considerable time needed for solving can thus be reduced.

5. AN ALGORITHM FOR SOLVING

In this section the algorithm to solve the introduced problem is described. It is built on the basis of the lemmas presented in Section 4.

Algorithm for finding the optimal solution – Branch-and-Bound with Restriction (BBR)

Denote: $p_0 = \min\{p_i\}_{i=1,..,n}$, $p_{min}=\min\{p_i | h^i_{x_i} < a^i_k\}$, $a_{min}=\min\{a^i_k\}_{i=1,..,n}$, *num_low_value* = the number of interval where $h^i_{x_i} \leq a_{min}/2$, *level* is the number of the first selected seller (to avoid redundant search), $L=\sum_{i=1}^n l^i_x$, $H=\sum_{i=1}^n h^i_{x_i}$, $F_{opt}=\min\{F\}$, $\forall i\in[1,n]$, $\Omega_i = \{[l^i_j,h^i_j]\}_{j=1,..,k}$, i.e. a set of intervals in which value q_i could be.

```
1.  initially, $L=0, H=0$, $p_0=\infty$, $p_{min}=\infty$, $F_{opt}=\infty$,
    num_low_value=0, level =1,
2.  i=level,
3.  if $\Omega_i\neq\{\varnothing\}$, take any interval $[l^i_{x_i}, h^i_{x_i}]$ from $\Omega_i$
    and remove it from $\Omega_i$, otherwise i=i+1, if i<n,
4.  update num_low_value; if num_low_value>1,
    then i=i-1 and return to 3,
5.  update L and H, $p_0$ and $p_{min}$,
6.  check the validation of (14)
    a. if $L\leq Q\leq H$ → go to 8,
    b. if $H<Q$ → i=i+1 and return to 3,
    c. if $L>Q$ → i=i–1and return to 3,
7.  check the conditions from Lemmas 3 and 4
    a. if all conditions are satisfied → go to 8,
    b. if not, i=i–1 and return to 3,
8.  apply the approach presented in Theorem 1 to
    find the final solution; update value F,
9.  if i>level, i=i -1 and return to 3,
10. if i=level and $\Omega_i = \{\varnothing\}$, then level = level+1, and
    return to 2, until level=n.
```

As described in the algorithm's label, the proposed algorithm is built in the branch-and-bound principle. Before realizing the final phase – finding values of quantities with given domains, a number of conditions are checked. The purpose is to avoid exploring useless cases. After any solution satisfying all necessary conditions presented in Lemmas 3 and 4 is achieved, the solver returns to the previous state and takes another unselected one to continue.

6. PERFORMANCE ANALYSIS

The algorithm presented above can be used to find the optimal solution. Despite many efforts to reduce the search space, the number of configurations to be examined is large. The kernel of the algorithm is Step 8, which is solvable in O(n*log$n$). Consequences 1 and 2 allow omitting a lot of small intervals from seller's offers. Lemmas 3 and 4 are used to verify the optimality conditions of final solutions before realizing Step 8. Many candidate configurations are reduced because of the conditions derived in Lemma 3 and 4. On the other hand, the complexity of the algorithm is not reduced to linear or pseudo-linear as wished. When the number of interval in seller's offers is large, the solving process is lengthy due to a large number of configurations. For that reason,

some restrictions have to be added in order to achieve the sub-optimal solution in acceptable time.
The above algorithm will be compared with some other algorithms which can be used to find the optimal solution too.

7. α-PRICE SOLUTION

One of the ways how to make the solving process faster is to reduce the dimension of sets $\Omega_i|_{i=1,..,n}$, i.e. the sets of intervals in which the values $q_i|_{i=1,..,n}$ could be. Let us denote $\Omega_i^\alpha=\{[l^i_j,h^i_j]\}_{j=s,..,k}$, where $p^i_s\leq\alpha$, i.e. a set of intervals in which the unit price p_i is not higher than the constant α. The minimal practical value that the constant α could have $\min\{p^i_k\}$. The achieved solution is called *α-price* solution, concerning the fact that unit price provided by every seller is not higher than α. Based on this discussion, the following modified algorithm is presented.

Algorithm for finding α-price solution - Branch-and-Bound with Restriction 2 (BBR-2):

1. initially, set $\alpha=\min\{p^i_k\}$, θ is any constant,
2. calculate all sets $\Omega_i^\alpha|_{i=1,..,n}$
3. call BBR to solve, but all original sets $\Omega_i|_{i=1,..,n}$ are replaced by $\Omega_i^\alpha|_{i=1,..,n}$
4. if any solution is achieved → STOP; otherwise, $\alpha=\alpha+\theta$ and return to 2.

The difference between BBR and BBR-2 is that smaller sets $\Omega_i^\alpha|_{i=1,..,n}$ are used to construct the final solution. The achieved solution is sub-optimal; however, its quality could be assessed by the following theorems.

Theorem 2: Let F_{opt} and F_α denote the total payment of the optimal solution and the solution achieved by applying BBR-2 with constant α; then

$$F_\alpha/ F_{opt} \leq\alpha/\min\{p^i_k|_{i=1,..,n}\} \qquad (27)$$

Proof: ∀i, $p_i\leq\alpha$, due to the assumption of finding *α-price solution*. Consequently, $F_\alpha\leq\alpha Q$. On the other hand, ∀i, $p_i \geq \min\{p^i_k|_{i=1,..,n}\}$, and then
$F_{opt}\geq Q*\min\{p^i_k|_{i=1,..,n}\}$ =>
$F_{opt}/\min\{p^i_k|_{i=1,..,n}\}\geq Q\geq F_\alpha/\alpha$. It is easy to derive (27).
The theorem is proved.

Theorem 3: BBR-2 is solvable in O(*n**log*k*)+ O(*n**log*n*).

Proof: ∀i, p^i_j increases when *j* moves from 1 to *k*; thus, let us identify maximal coefficient *s*, so that $p^i_s\leq\alpha<p^i_{s-1}$ is achievable in O(log*k*). Calculating all sets $\Omega_i^\alpha|_{i=1,..,n}$ requires maximally O(*n**log*k*) operations. The existence of any α-price solution is checked by using (14), namely by checking the condition

$$\sum_{i=1}^n a^i_{s_i} \leq Q\leq \sum_{i=1}^n a^i_k \text{ , where } \forall i,\ \Omega_i\neq\{\varnothing\} \qquad (28)$$

where s_i is the lowest index, which satisfies $p^i_{s_i}\leq\alpha$. This operation is solvable in O(*n*). When any solution is achieved, the algorithm stops; in addition, according to Theorem 1, Step 4 is solvable in O(*n**log*n*); therefore the BBR-2 algorithm achieves any α-price solution in O(*n**log*k*)+ O(*n**log*n*).

8. SIMULATION RESULTS AND DISCUSSION

Both described algorithms have been implemented in C++ and verified in many examples with real data. All experiments have been tested in the same PC with processor P3, 766 MHz. To verify the performance of the described algorithms, the A^* search algorithm [Kumar et al., 1994] is chosen to compare with BBR. A^* is selected because it can be used to find the optimal solution. Another algorithm presented in [Dang & Jennings, 2002] with the purpose to find sub-optimal solution is selected to compare with BBR-2. This algorithm is based on the principle similar to the best-first search algorithm [Kumar et al., 1994], so we will denote it as BFS.
In experiments, the number of sellers is set by 50, and the number of intervals in each offer is from 20 to 40, ($k\in[20,40]$). Each seller can offer maximally from 10^4 up to $4*10^4$ units (a^i_k is randomly selected from $[10^4, 4*10^4]$) and the demand $Q\in[10^4,5*10^5]$. The simulation results are shown in Table 1.
In Table1, Q represents the desired demand; next two columns show the performance of BBR and A^* algorithm in seconds. F_{opt} is the total payment of the optimal solution (the really lowest payment); $\lambda=\max\{F_{sub}/F_{opt}\}$ is the coefficient used to predict the area where the optimal solution could be, based on the results achieved by applying each proposed algorithm. F_α denotes the results of BBR-2. Columns F_{sub}/F_{opt} (%) and F_α/F_{opt} (%) express the ratio between the sub-optimal and the optimal solution in percentage. Running time of both the algorithms BBR-2 and BFS are shown in the columns “running time” and they are given in milliseconds.
The achieved results point out that BBR is applicable for situations when a small number of sellers can satisfy the desired demand. Using the derived theoretical results reduces the solution space several times; BBR is about 10 times faster than A^* algorithm, however, it is not sufficient for more complicated situations. For larger cases, in which the solver has to explore configurations consisting of more than 10 from 50 existing sellers, in order, BBR seems not to be a practical approach to get the desired demand.
On the other hand, BBR-2 and BFS achieved of average quality solutions, but required only little time for realization. The sub-optimal solutions achieve about 102% of the really optimal one. In many experiments BBR-2 achieved better results than BFS, mainly in such cases when seller’s offers have significantly different distribution of unit prices and lengths of intervals. When seller’s offers are close one to another (i.e. the distribution of unit prices and the dimension of intervals are relatively equal in each offer), both algorithms achieved equivalent results. BFS is somewhat faster, because it always takes the best branch to continue the search. On the other

hand, it has to be mentioned that BBR-2 requires a lot of time to generate sets $\{\Omega_i\}_{i=1,..,n}$ and it explores many redundant configurations that were not solutions in previous cycles. To avoid redundant exploring, each examined configuration could be indicated, but this requires considerable memory for storage. BBR-2 is clearly better than BFS regarding the prediction of the range of the optimal solution. BFS predicts that the optimal solution might have F value in the range of: $[F_{sub}/50, F_{opt}]$, (50 is the number of sellers); to compare, BBR-2 predicts more precisely the range of the optimal solutions' value. BBR-2 prediction is between $[F_{sub}/1{,}1, F_{sub}]$ bounds. From practical point of view, applying BBR-2 gives a better chance to assess the total payment that the buyer minimally can pay; it is an important factor in decision making during negotiation when the time for calculating is limited.

To compare BBR-2 and BFS, a number of experiments were made. The achieved results confirmed that both algorithms are comparable as to the achieved solution. However, BBR-2 is better when sellers have different supplies with different distribution of unit prices. In Table 2 some selected results are given, which show the differences between both algorithms applied for the above mentioned cases. The coefficient of prediction that BBR-2 can achieve is much better than BFS. That means the actually optimal solution is very close to the solution achieved by applying BBR-2.

Table 2: comparison between BBR-2 and BFS

Q (10^4)	BBR 2			BFS		
	F_α/F_{opt} (%)	λ	run time (ms)	F_{sub}/F_{opt} (%)	λ	run time (ms)
2,776	100,00	1,03	65	101,42	50	20
3,689	100,11	1,03	71	100,65	50	14
4,592	100,18	1,03	51	100,52	50	14
5,489	100,07	1,03	84	100,14	50	19
6,821	100,00	1,00	34	101,04	50	23
7,278	100,40	1,03	108	100,34	50	18

Although the real number of suppliers might not be as high as assumed in simulated experiments, but applying BBR and BBR-2 allows predicting the minimal cost of the goods what the manufactory should pay. That result allows minimizing the production cost of products, and increases the manufactory's concurrent abilities.

9. CONCLUSSION

In this paper, two algorithms (BBR and BBR-2) have been presented to solve the "reverse auction" problem with single item for trading. In the BBR algorithm, the solution space is reduced many times by using consequences of a number of derived theoretical results. The global solution is achievable for some classes of cases in acceptable time. For more complicated cases, the heuristic BBR-2 algorithm is proposed. It is a simpler version of BBR when only a part of seller's offers is considered for calculation. The BBR-2 algorithm has pseudo-linear complexity and it is realizable for large cases with many sellers and the desired demand. In addition, it has much better predicted range of the optimal solution's value than the BFS algorithm.

The objective for our future research is to extend the BBR and BBR-2 algorithm to solve the "reverse auction" problem with multi-item trading.

Acknowledgement: This paper is partially supported by APVT and VEGA grant agencies under grants No APVT 51 011602 and VEGA 2/1101/21.

REFERENCES

Dang, V. D. and Jennings, N. R. (2002): Polynomial algorithms for clearing multi-unit single item and multi-unit combinatorial reverse auctions. *ECAI-2002,* Lyon, France, 23-27.

Gonen R. and Lehmann D. (2000): Optimal solutions for multi-unit combinatorial auctions: Branch and bound heuristics. In *ACM Conference on Electronic Commerce,* Minneapolis, MN, 2000, 13–20.

Kumar V., Grama A., Gupta A., and Karypis A. (1994): Introduction to Parallel Computing. Design and Analysis of Algorithms. *The Benjamin/Cummings Publishing Company, Inc*., California, 1994. ISBN 0-8053-3170-0.

Sandholm, T., and Suri, S (2002): Optimal Clearing of Supply/Demand Curves. In *Proceedings of the 13th Annual Int. Symposium on Algorithms and Computation (ISAAC)*, Vancouver, Canada, 2002.

Sandholm T., Suri S., Gilpin A., Levine D. (2002): Winner determination in combinatorial auction generalizations. In: *Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS)*, Bologna, Italy, 2002, 69–76.

Table 1: Simulation results and comparison between BBR and A^*, BBR-2 and BFS.

Q (10^4)	BBR - running time (s)	A^* - running time (s)	F_{opt} (monetary unit)	BBR 2			BFS		
				F_α/F_{opt} (%)	λ	Running time (ms)	F_{sub}/F_{opt} (%)	λ	running time (ms)
1,5	0,6	0,8	112500	101,33%	1,06	118	100,98%	50	18
2	0,217	0,46	143800	100,14%	1,00	16	100,14%	50	10
2,5	7,16	12,45	184500	100,49%	1,06	103	100,16%	50	15
3	6,38	16,71	214800	100,56%	1,00	30	100,56%	50	8
4	26,89	56,87	286400	100,56%	1,00	23	100,56%	50	6
5	132,51	2156,9	360000	100,00%	1,03	80	100,00%	50	14
6	547,95	4577,4	431200	100,19%	1,00	62	100,19%	50	11
8	1882,1	13543,8	574400	100,28%	1,06	105	100,28%	50	10
10	14413,5	232561,8	720000	100,00%	1,06	142	100,00%	50	29