2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Distributed Algorithms for Biobjective Assignment Problems

Chendong Li, Chulwoo Park, Krishna R. Pattipati, *Fellow, IEEE*, and David L. Kleinman, *Fellow, IEEE*

Abstract— **In this paper, we study the biobjective assignment problem, a NP-hard version of the classical assignment problem. We employ an effective two-phase method with certain enhancements: in Phase I, we use a distributed auction algorithm to solve the single objective assignment problems to obtain the so-called supported Pareto optimal solutions; we apply a ranking approach with tight upper/lower bounds in Phase II to obtain the non-supported Pareto optimal solutions. Moreover, a randomized algorithm for Phase II is proposed that supports finding the approximation on a polynomial time basis. Extensive experiments are conducted using SGI Altix 3700 and computational results are reported based on a large set of randomly generated problem instances. Also, some experimental results of the distributed auction algorithm on large data-size assignment problems are provided.**

I. INTRODUCTION

A. Motivation

This research is motivated by the mission planning and monitoring activities associated with the Navy's maritime operations center (MOC), in which multiple decision makers (DMs) with partial information and partial control over assets are involved in the development of operational level plans. The MOC emphasizes networked distributed planning capabilities, and decentralized execution for assessing, planning and executing missions across a range of operations [8].

Motivated by the distributed planning problems and asset-task allocation in large-scale organizations, where information processing and decision making are distributed among DMs, a novel variation of assignment problem, wherein each of DMs knows only a part of the weight matrix and/or controls a subset of the assets, is introduced in [5]. Furthermore, the auction algorithms were extended to realistic settings with various information, communication and organizational structures to quantify the impact of structures on planning delays [6] [7]. In this paper, we apply distributed auction algorithm to solve the biobjective assignment problem in the same context.

The primary focus of a single objective optimization problem is to find the global optimum, which achieves the best objective function value. However, real-world scenarios usually involve more than one objective, which might conflict with each other. In the context of mission planning, a commander needs to trade-off task accuracy, asset usage cost, and mission risk. In this scenario, instead of a global optimum, we have a set of alternative trade-offs, termed the Pareto front. In this paper, we are exploring how the distributed auction algorithm can cope with biobjective functions, i.e., reward (benefit, accuracy) and risk, in the context of an assignment problem.

This work was supported by the U.S. Office of Naval Research under Grant N00014-09-1-0062.

C. Li is with the Computer Science Engineering Department, University of Connecticut, Storrs. C. Park and K. R. Pattipati are with the Electrical Engineering Department, University of Connecticut, Storrs, CT 06029, USA {chendong.li; chp06004; krishna}@engr.uconn.edu

D. L. Kleinman is with the Department of Information Sciences, Naval Postgraduate School, Monterey, CA 93943, USA dlkleinm@nps.edu

B. Background on the Assignment Problem

As a fundamental combinatorial optimization problem in operations research, the assignment problem (AP) appears as a subproblem of the transportation problem [34], minimum cost flow problem [3] and the traveling salesman problem [23]. There are specialized efficient algorithms to solve the assignment problem, such as the well-known Hungarian method [30] (an updated version as [31]) and the auction algorithm [12] [13] [21], which are designed to make use of the particular structure of APs. Moreover, forward and reverse auction algorithms were developed in [15] and they were extended to solve the transportation problem [15], the shortest path problem [19] and the minimum cost flow problem [16] [17] [18]. Early parallelized versions of these specialized algorithms for the assignment problem are presented in [14] [22] [20].

Here, we consider a simple extension of the AP, viz., the biobjective assignment problem (BAP). Solving the BAPs efficiently is of significant importance in that it is the simplest version of the more general multi-objective assignment problem. Unfortunately, this problem is NP-hard [25].

The rest of the paper is organized as follows: Section II provides a literature review. Section III presents the problem formulation of BAPs as well as the two-phase algorithm. The distributed auction algorithm is described in Section IV and computational results are reported in Section V. Section VI concludes with a summary and future research directions.

II. LITERATURE REVIEW

In the early stages of research, a majority of papers on multi-objective assignment problems sought to identify the supported efficient solutions (as shown in Fig. 1) [37] by utilizing convex combinations of the objective functions or via goal programming [44]. The non-supported efficient solutions are usually ignored in the computation. In order to compute the non-supported efficient solutions, one employs branch-and-bound methods, such as in [29] and [39]. Meta-heuristics, such as simulated annealing [45] [24], tabu search [28], and genetic algorithm [11] have also been proposed.

Ulungu and Teghem [26] described the first exact and complete solution approach for the BAP by employing a two-phase method: in Phase I, it computes the complete set of

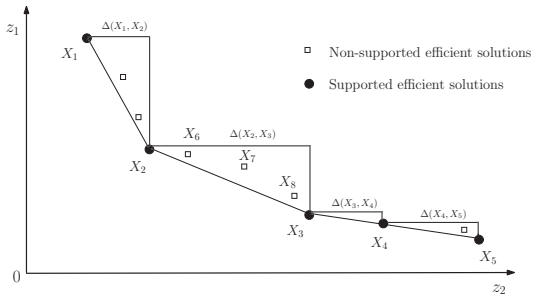

Fig. 1. Supported and non-supported efficient solutions in the objective space $\mathcal{Z}$.

all the supported efficient solutions; in Phase II, it computes the complete set of non-supported efficient solutions (refer to section II for details). In [35], this method was employed to solve the biobjective knapsack problems. Tuyttens *et al.* implemented this two-phase method and reported the results in [10]. Degoutin and Gandibleux did similar experiments by using a mixed integer programming (MIP) solver and reported their empirical results in [27]. An improvement to this method was presented in [1] by utilizing a ranking approach in Phase II. A population-based heuristic using path relinking was proposed in [38]. Later, a more comprehensive and improved two-phase method was proposed in [9], which is, to our best knowledge, the state-of-the-art algorithm for identifying the exact and complete set of the supported and non-supported efficient solutions.

The two-phase method can efficiently solve not only the BAPs [36], but also a large number of combinatorial optimization problems, such as biobjective network flow [3], knapsack [35], spanning tree [43], and traveling salesman problems [23]. A comprehensive survey of these problems are given in [40]. The basic idea here is to solve the single objective problem with efficient algorithms to obtain supported efficient solutions in Phase I, and enumerate the non-supported efficient solutions in Phase II. Moreover, the two-phase method was recently extended to solve the three-objective assignment problem [2]. Some papers also focus on identifying the feasible solutions of the multi-objective assignment problem, such as [4].

III. BIOBJECTIVE ASSIGNMENT PROBLEM

The biobjective assignment problem can be described as one of assigning n workers to m jobs or assigning n assets to m tasks with the minimum overall cost and the maximum benefits (or minimum negative benefits). If m equals n, the problem is called a symmetric assignment problem and this is the problem considered here.

A. Problem Formulation

Generally, the biobjective assignment problem can be formulated as follows:

$$
\begin{aligned}
\text{Minimize} \quad z_1: \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^1 x_{ij} \\
z_2: \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^2 x_{ij} \\
\text{Subject to} \quad & \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, 2, ..., n \\
& \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, 2, ..., n \\
& x_{ij} \in \{0, 1\}, \quad i, j = 1, 2, ..., n
\end{aligned}
$$

where c^1 and c^2 are the coefficient matrices composed of integers and x is the matrix of decision variables. Each entry x_{ij} is a binary variable: x_{ij} equals one if i is assigned to j and equals zero otherwise. To simplify the representation, we introduce the following notation:

$$
\begin{aligned}
\text{Min} \quad z_1: \quad & c^1 x \\
z_2: \quad & c^2 x \\
\text{s.t.} \quad & x \in \mathcal{X}
\end{aligned}
$$

where x is the decision variable and $\mathcal{X}$ is called the *decision space* or *solution space*. Similarly, the *objective space* or *criterion space*, denoted by $\mathcal{Z}$, can be represented as follows:

$\mathcal{Z} = \{(z_1, z_2) | z_1 = c^1 x, z_2 = c^2 x, x \in \mathcal{X}\}$.

A *feasible* solution is a solution that satisfies all the constraints of the BAP. A feasible solution x^* is called *efficient* if no other feasible solution x' can be found in the decision space that satisfies the following conditions:

$z_1(x') \leq z_1(x^*)$; $z_2(x') \leq z_2(x^*)$, with at least one inequality strictly holding.

$z(x^*)$ is called a non-dominated Pareto point. In the objective space, the set of *efficient* solutions $\mathcal{X}_e$ can be expressed as $\mathcal{X}_e = \{x \in \mathcal{X}_f \mid \nexists x' \in \mathcal{X}_f, z_1(x') < z_1(x) \text{ and } z_2(x') < z_2(x)\}$, where $\mathcal{X}_f$ is the set of *feasible* solutions. The set of efficient solutions $\mathcal{X}_e$ can be mapped to the non-dominated set $\mathcal{Z}_{nd}$ in the objective space $\mathcal{Z}$, where

$\mathcal{Z}_{nd} = \{(z_1, z_2) | z_1 = c^1 x, z_2 = c^2 x, x \in \mathcal{X}_e\}$.

The set of non-dominated points can be grouped into supported and non-supported points. Correspondingly, the set of efficient solutions can be partitioned into supported and non-supported efficient solutions, where supported efficient solutions are optimal solutions of the weighted sum single objective problems. Solutions x, x^* are said to be *equivalent* if $z_k(x) = z_k(x^*)$ where $k = 1, 2$.

Here is a brief summary of notation in the rest paper.

- The set of efficient solutions is denoted by $\mathcal{X}_e$ and its image in $\mathcal{Z}$ is the non-dominated frontier $\mathcal{Z}_{nf}$.
- Supported efficient solutions $\mathcal{X}_{se}$ are optimal solutions of weighted sum of single objective assignment problems. Non-supported efficient solutions are efficient solutions that are not optimal for BAP$_\lambda$ ($\lambda > 0$).
- Supported extreme solutions are efficient solutions with (z_1, z_2) located on extreme points of the vertex set of $\mathcal{Z}$. Supported non-extreme efficient solutions (z_1, z_2) are

not located on the extreme point of vertex set of $\mathcal{Z}$. As shown in Fig. 1, X_1, X_2, X_3, X_5 are supported extreme solutions while X_4 is a supported non-extreme efficient solution because it is located on the line segment joining X_3 and X_5.

In general, the biobjective assignment problem is NP-hard [42] [32] because an exponential number of optimal solutions may exist. The process of solving the BAP is to compute the complete set of supported and non-supported efficient solutions, which are identified separately with the two-phase algorithm.

B. The Two-phase Algorithm

The general idea of the two-phase algorithm is to compute the set of supported efficient solutions in Phase I, and in Phase II employ an enumeration procedure to compute the set of non-supported efficient solutions with the information obtained from supported efficient solutions identified by Phase I.

There are two important steps in Phase I: First computing the lexicographical efficient solutions (as shown in line 1 and 2 in algorithm 1). In lines 1 and 2, the algorithm solves the assignment problem with the auction algorithm, which is easy to parallelize. The second step, as shown in lines 9 to 18 in algorithm 1, is to identify the complete set of supported solutions with a search procedure, while lines 3 to 8 setup the search procedure in the while loop.

The weighted sum of single objective assignment problems can be expressed as $z_\lambda(x) = (\lambda c^1 + c^2)x$ where $\lambda \in \mathcal{R}^+$ and λ is the slope of the line determined by two efficient solutions (z_1, z_2) and (z'_1, z'_2) in the objective space $\mathcal{Z}$. This expression is used to compute the new λ when two efficient solutions are known, as shown in line 10 in algorithm 1.

Algorithm 1 Phase_One()

Require: c^1, c^2

1: $z^u = auction\_Solve(c^1)$; // x^u is optimal solution
2: $z^l = auction\_Solve(c^2)$; // x^l is optimal solution
3: **if** $z^u = z^l$ **then**
4: **return** 0;
5: **else**
6: $S = \{z^u, z^l\}$;
7: **end if**
8: Let $z^+ = z^u, z^- = z^l$;
9: **while** $z^+ \neq z^-$ **do**
10: $\lambda = \lambda(z^+, z^-)$; //update λ
11: $z^* = auction\_Solve(\lambda c^1)$; //$x^*$ is optimal solution
12: **if** $z_\lambda(x^*) < \lambda z_1^+ + z_2^+$ **then**
13: $S = S \cup \{z^*\}$;
14: **else**
15: $z^+ = z^-$;
16: $z^- = S.next(z^+)$;
17: **end if**
18: **end while**

The two-phase algorithm determines the supported efficient solutions first and then computes the non-supported efficient solutions in Phase II. The main idea in Phase II is to make use of the supported efficient solutions identified in Phase I to reduce the search space where the non-supported efficient solutions may exist. Specifically, Phase II explores the triangles defined by two consecutive supported efficient solutions in the objective space $\mathcal{Z}$. As shown in Fig. 1, the triangles $\Delta(X_1, X_2), \Delta(X_2, X_3), \Delta(X_3, X_4)$, and $\Delta(X_4, X_5)$ are explored, respectively, in Phase II. To search for the non-supported efficient solutions in an effective manner, Lower Bound (LB) and Upper Bound (UB) and heuristics are employed to reduce the search space during the enumeration process. A ranking approach is usually used to achieve better performance [1] [9]. The recent paper [9], which utilizes the two-phase method, outperforms the algorithms presented in [1]. One of the important reasons is that the algorithm in [9] applies a tight upper bound proposed in [1] and keeps on updating the lower bound within each iteration of Phase II.

Compared to Phase I, Phase II is a more complicated procedure because it explores all the triangles defined by two consecutive supported efficient solutions to obtain the non-dominated supported points located inside the triangle Δ. The original method used the LB and UB as well as the variable fixing method to enumerate all the possible assignments. Recent papers, such as [26] [10] [1], have presented tightened UB and LB to speedup the Phase II solution process.

Formally, let x^+, x^- be the two consecutive supported efficient solutions that correspond to z^+ and z^-, specifically (z_1^+, z_2^+) and (z_1^-, z_2^-) in the objective space $\mathcal{Z}$. As in Phase I, λ is the weight for which both x^+ and x^- are optimal solutions of the weighted sum, i.e., BAP_λ. To simplify, we use $\Delta(z^+, z^-)$ to denote the interior of the triangle of z^+ and z^- in the objective space $\mathcal{Z}$.

The essential part of Phase II is the while loop shown from lines 5 to 17 in Algorithm 2, which employs two procedures to compute the non-supported efficient solutions inside the $\Delta(z^+, z^-)$. The first procedure is $Kbest()$ (line 6), which is to compute the K^{th} best solution z^K ([9] provides a detailed process for computing the K^{th} best solution). If it is not dominated by other points in the current efficient solution set, it will be added to the set (lines 8 and 9). Another important procedure is line 10, which is to update the Upper Bound (UB). This UB is first proposed in [1], and rephrased in [9].

IV. THE DISTRIBUTED AUCTION ALGORITHM

In the early parallelization of the auction algorithm, each processor adjusts its own dual prices on the basis of local information communicated by adjacent nodes and these implementations did not provide large speedups [16]. Multiple bids were carried out in parallel in the later parallelization algorithms, while the calculation of each bid was shared among several processors, which were able to achieve better speedups [14]. Recently, different distributed auction algorithms were proposed in [5] [41], where [5] assumes that each decision maker only knows part of the benefit matrix and needs to coordinate with the rest of the decision makers to arrive at a globally optimal assignment.

Algorithm 2 Phase_Two()

Require: z^+, z^-

$\lambda = \lambda(z^+, z^-)$;
$S = \{z^u, z^l\}$;
$UB = \max\{\lambda z^+ + z^-, \lambda z^- + z^+\}$;
$LB = \lambda z_1^+ + z_2^+; K = 1$;
while $LB \leq UB$ **do**
  $z^K = Kbest(K, \lambda)$;
  if Non_Dominated(z^K) **then**
    $S = S \cup \{z^K\}$;
    for $i = 1...q-1$ where $z^q = z^-$ **do**
      $UB = \max\{\lambda(z_1^{i+1} - 1) + (z_2^i - 1)\}$
    end for
  else {Dominated case}
    do nothing;
  end if
  $LB = \lambda z_1^K + z_2^K$; //update LB
  $K = K + 1$;
end while

To achieve the speedup in the solution process of BAPs, we utilize the distributed auction algorithm to solve the single objective assignment problem. Specifically, in Phase I, we employ the auction algorithm to solve the lexicographical and weighted sum of the BAP. In the following section, we first provide a brief description of coordination structures for the distributed auction algorithm and then present the distributed auction algorithm in detail. To achieve further speedup in Phase II, we propose a randomized algorithm.

A. The Coordination Structures

1) Information Structure: Based on the computational results reported in [5], here we consider the horizontal information structure only. In this structure, each decision maker only knows certain rows of the cost and benefit matrix corresponding to a set of tasks. Also, the decision makers cannot exchange information with each other. The information structure for the normal sequential assignment problem is termed the centralized information structure.

2) Communication Structure: We consider blackboard communication structure wherein decision makers send their bids, as well as the best and the second best profits to the blackboard; decision makers may choose to update their bids after observing the bids on the blackboard.

3) Organization Structure: We consider a parallel structure, where there is one root decision maker who supervises and coordinates the rest of the decision makers.

B. Distributed Auction Algorithm for the Single Objective Assignment Problems

The distributed forward auction with horizontal information structure can be described as follows: for each task, each decision maker (DM) finds the best asset, best profit and the 2nd best profit. Then, for each task from the task set, all DMs send their bids to the blackboard. The coordinator (blackboard) assigns an asset to certain task to attain the maximum and posts the bid back to the blackboard. In the following, each DM updates the bid based on the bid on the blackboard. The pseudo code of this algorithm is given in Algorithm 3.

Algorithm 3 auction_Solve(c)

Require: c, BB

//each DM bids for his tasks Given DM k, each task i_k
for each task $i \in T$ **do**
  find the best asset, best and 2nd best profit $j_{i_k} = arg\max\{c_{i,j} - p_j\}$;
  compute bid;
end for
for each task i $\in T$ **do**
  for each DM $D_k \in K$ **do**
    send(bid, BB) //their bids are sent to the blackboard BB
  end for
end for
the coordinator assigns an asset j to task i attaining the maximum p_j and posts the bid back to the blackboard.
each DM updates the bid based on the best bid on the blackboard.

C. Randomized Algorithm for Phase II

To speedup Phase II, we propose a randomized algorithm for Phase II. The intuition of the randomized algorithm comes from the Pareto optimal front. To make this frontier convex, after computing the supported efficient solutions in Phase I and instead of going through Phase II, the algorithm randomly picks one of the successive assignments from the supported efficient solutions. This yields efficient solutions without exploring the inside of the triangle formed by the two consecutive supported efficient solutions. A significant advantage of this randomized algorithm is that it could guarantee a better optimal value than the non-supported efficient solutions identified in Phase II because the objective values (z_1, z_2) are located on the convex frontier in the objective space $\mathcal{Z}$. In the context of military planning, this also provides an element of surprise and unpredictability.

Specifically, given probability $P_1 \in [0, 1]$, for the two consecutive supported efficient solutions (x_l, x_{l+1}) (where $l = 0, 1, 2...|\mathcal{X}_e|$), we can interpret the efficient solutions $P_1 \times x_l + (1 - P_1) \times x_{l+1}$, as the linear combination of the two consecutive supported efficient solutions (x_l, x_{l+1}). Similarly, the corresponding points in the objective space $\mathcal{Z}$ can be expressed as $\mathcal{Z}' = \{(z_1', z_2')|z_1' = P_1 \times z_1^l + (1 - P_1) \times z_{l+1}, z_2' = P_1 \times z_2^l + (1 - P_1) \times z_2^{l+1}, l = 0, 1, ..|\mathcal{X}_e|\}$.

The randomized process could be interpreted as follows: Suppose in 1000 executions of the biobjective assignment problem, the team chooses x_l $1000 \times P_1$ times and x_{l+1} $1000 \times (1 - P_1)$ times, where $l = 0, 1, ..|\mathcal{X}_e|$ and x_l, x_{l+1} denote the consecutive supported efficient solutions from the set $\mathcal{X}_e$ obtained from Phase I. Evidently, the randomized solutions are on the supported Pareto front.

V. COMPUTATIONAL RESULTS

A. Experimental Setup

The algorithms are implemented in C++ with MPI (Message Passing Interface) library and tested in a Unix environment on the SGI Altix 3700 (sgi1.engr.uconn.edu). The program is compiled using `icc` version 9.0 with the arguments `-lrt` and `-lmpi`. Each processor is from GenuineIntel Itanium 2 family with 1500 MHz. This shared memory supercomputer is configured with 60G RAM. The source code package is composed of three parts:

- Part 1: The matrix generator. It generates the random matrix given the size of the matrix and the range of the elements of the matrix (typically [100, 1000]).
- Part 2: The distributed auction algorithm in a master-slave paradigm with 1 master process and $(n-1)$ slave processes in an n-processor system.
- Part 3: The two-phase algorithm and the randomized algorithm.

B. Computational Results on SGI Altix 3700

1) Results for Distributed Auction Algorithm: Speedup and efficiency are normally used to evaluate the parallelization process. Speedup is equal to $T_{sequential}/T_{parallel}$, where $T_{sequential}$ and $T_{parallel}$ denote the execution time of the sequential and parallel algorithm respectively. Efficiency is used to estimate how well-utilized the processors are in computing the results, which is compared to how much time is used in communication and synchronization. It can be calculated by $Speedup/Proc$ where $Proc$ denotes the number of processors used in parallel computing.

TABLE I
PERFORMANCE OF THE DISTRIBUTED AUCTION ALGORITHM

CPU time in Milliseconds					
Problem	#Run	Sequential	Distributed	Speedup	Efficiency
API-10	1	0.501	0.481	1.042	26.04%
	2	0.546	0.473	1.154	28.86%
	3	0.618	0.454	1.361	34.03%
API-50	1	0.776	0.613	1.266	31.65%
	2	0.803	0.562	1.429	35.72%
	3	0.769	0.524	1.468	36.70%
API-100	1	19.176	6.496	2.964	74.12%
	2	19.140	6.348	3.015	75.38%
	3	19.114	6.373	2.999	74.98%
API-500	1	482.193	214.296	2.250	56.25%
	2	479.933	218.468	2.197	54.92%
	3	479.501	202.510	2.368	59.19%
API-1000	1	1917.294	790.498	2.425	60.64%
	2	1920.200	801.553	2.396	59.89%
	3	1922.038	813.919	2.361	59.04%
API-3000	1	18997.489	7397.399	2.568	64.20%
	2	18418.097	7140.541	2.579	64.48%
	3	19054.572	7937.026	2.403	60.07%
API-5000	1	59739.896	20447.697	2.922	73.04%
	2	60326.417	20521.972	2.940	73.49%
	3	64561.193	20710.660	3.117	77.93%

Table I presents the detailed simulation results, including speedup and efficiency on problem sizes ranging from 10 $\times$ 10 to 5000 $\times$ 5000. The distributed computation time in Table I is obtained by using 4 processors on SGI Altix 3700. As shown in Table I, the problem instance is termed "API-XX", where "API" stands for the assignment problem instance and "XX" denotes the dimension of the matrix. In Table I, the efficiency of the processors is slowly increasing with the growth of problem size.

2) Distributed Auction Algorithm with Two-phase Algorithm: This section presents the preliminary computational results for the proposed methods. The BAP problem instance is generated randomly given the dimension of the matrix and the interval of its elements. The range of each element of the matrix is [10, 300] and the time unit is in milliseconds.

Randomized assignment results are also presented in the table. The problems in Table II are named in the form of "BAPI-XX", where "BAPI" denotes the biobjective assignment problem instance and "XX" denotes the data size, which is the dimension of the cost matrices C_1 and C_2 in the biobjective assignment problem. Table II shows the significant speedup of the randomized algorithm with the increase in problem size.

TABLE II
PRELIMINARY RESULTS FOR THE BAPS

CPU time in Milliseconds				
Problem	#Run	Two-phase	Randomized	Speedup
BAPI-10	1	0.906	0.714	1.269
	2	0.694	0.630	1.012
	3	0.792	0.425	1.864
BAPI-30	1	147.276	0.832	177.014
	2	144.658	0.944	153.239
	3	140.282	0.746	188.046
BAPI-50	1	2305.643	60.728	37.967
	2	2285.278	61.330	37.262
	3	2280.254	61.972	36.795
BAPI-70	1	13840.763	192.213	72.007
	2	13473.118	186.032	72.424
	3	13980.725	196.477	71.157
BAPI-90	1	30761.456	320.195	96.071
	2	30860.169	286.217	107.821
	3	30980.364	300.702	103.027

VI. CONCLUSIONS AND FUTURE WORK

We present an enhanced two-phase algorithm to compute the complete set of efficient solutions for biobjective assignment problems. Specifically, our approach employs the effective two-phase method with certain enhancements: in Phase I, we use a distributed auction algorithm to solve the single objective assignment problem, and in Phase II, apply a ranking approach with tight upper/lower bounds. Moreover, a randomized algorithm for Phase II is proposed to achieve further speedup. We conducted empirical studies on SGI Altix 3700 and reported computational results based on a set of randomly generated BAP instances. We also provided test results of the distributed auction algorithm for reasonably large-size assignment problems, i.e. 5000 $\times$ 5000.

Our future work seeks to generalize the enhanced two-phase method to solve the general multi-objective assignment

problems. Moreover, we could extend our methodology to solve general multi-objective combinatorial optimization problems, such as planning and scheduling problems, network flow problems, traveling salesman problems, and so forth.

REFERENCES

[1] A. Przybylskia, X. Gandibleuxa, M. Ehrgotta, "Two phase algorithms for the bi-objective assignment problem", *European Journal of Operational Research*, Vol. 185, Issue 2, pp. 509–533, March 2008.

[2] A. Przybylski, X. Gandibleux and M. Ehrgott, "A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives", *Discrete Optimization* 7(3): 149–165, 2010.

[3] A. Sedeo-Noda and C. Gonzlez-Martn, "An algorithm for the biobjective integer minimum cost flow problem", *Computers & Operations Research* 28 (2), pp. 139–156, 2001.

[4] A. Bufardi. "On the Efficiency of Feasible Solutions of a Multicriteria Assignment Problem". *The Open Operational Research Journal*, 2, pp. 25–28, 2008.

[5] C. Park, W. An, K. R. Pattipati, and D. L. Kleinman, "Quantifying the impact of information and communication structures via distributed auction algorithm". In proceedings of the *2010 IEEE International Conference on Systems Man and Cybernetics*, pp. 2200–2207, 2010.

[6] C. Park, W. An, K. R. Pattipati, and D. L. Kleinman, "Quantifying the impact of information and organizational structures via distributed auction algorithm: Blackboard communication structure", *IEEE Trans. Syst., Man, Cybern. A*, submitted for publication.

[7] C. Park, K. R. Pattipati, W. An, and D. L. Kleinman, "Quantifying the impact of information and organizational structures via distributed auction algorithm: Point-to-point communication structure", *IEEE Trans. Syst., Man, Cybern. A*, to be published.

[8] Commander United State Fleet Forces Command, "Maritime Headquarters with Maritime Operations Center Concept of Operations", Rev. A, Initial draft vol. 2, Jan. 2008.

[9] C. R. Pedersen, L. R. Nielsen, K. A. Andersen, "The Bicriterion Multimodal Assignment Problem: Introduction, Analysis, and Experimental Results", *INFORMS Journal on Computing*, Vol. 20, No. 3, pp. 400-411, Summer 2008.

[10] D. Tuyttens, J. Teghem, Ph. Fortemps and K. Van Nieuwenhuyse, "Performance of the MOSA method for the bicriteria assignment problem", *Journal of Heuristics*, 6, pp. 295-310, 2000.

[11] D. E. Goldberg, Genetic Algorithm in Search, Optimisation and Machine Learning, Reading, MA: Addison–Wesley, 1989.

[12] D. P. Bertsekas, "A Distributed algorithm for the assignment problem", Lab. for Information and Decision Systems, Working Paper, M.I.T., Cambridge, MA, March 1979.

[13] D. P. Bertsekas, "A New algorithm for the assignment problem", *Math. Programming*, vol. 21, pp. 152-171, 1981.

[14] D. P. Bertsekas and D. A. Castanon, "Parallel synchronous and asynchronous implementations of the auction algorithm", *Parallel Computing*, vol. 17, pp. 707-732, 1991.

[15] D. P. Bertsekas and D. A. Castanon, "The auction algorithm for transportation problems," *Analysis of Operation Research*, vol. 20, pp. 67-96, 1989.

[16] D. P. Bertsekas and J. Eckstein, "Distributed asynchronous relaxation methods for linear network flows problems," in Proc. *IFAC'87*, Munich, Germany, Jul. 1987.

[17] D. P. Bertsekas and D. A. Castanon, "The auction algorithm for minimum cost network flow problem," Lab. For Information and Decision Systems, Technical Report LIDS-P-1925, M.I.T., Cambridge, MA, 1989.

[18] D. P. Bertsekas and D. A. Castanon, "A generic auction algorithm for the minimum cost network flow problem," Lab. For Information and Decision Systems, M.I.T., Cambridge, MA, 1991.

[19] D. P. Bertsekas, "An Auction algorithm for shortest paths," *SIAM Journal on Optimization*, vol. 1, pp. 425-447, 1991.

[20] D. P. Bertsekas and D. A. Castanon, "Parallel Asynchronous Hungarian Methods for the Assignment Problem", *ORSA Journal on Computing*, Vol. 5, No. 3, Summer 1993.

[21] D. P. Bertsekas, "Auction Algorithms", *Encyclopedia of Optimization* pp. 128–132, 2009.

[22] E. Balas, D. Miller, J. Pekny, P. Toth, "A parallel shortest augmenting path algorithm for the assignment problem", *Journal of the ACM (JACM)*, Vol. 38, Issue 4, pp 985-1004, Oct. 1991.

[23] E. L. Lawler, "The quadratic assignment problem", *Management Science*, Vol. 9, No. 4, pp. 586–599, Jul. 1963.

[24] E. L. Ulungu, "Optimisation Combinatoire multicritère: Détermination de l'ensemble des solutions efficaces et méthodes intéractives", PhD thesis, Université de Mons-Hainault, Faculté des Sciences, 1993.

[25] E. L. Ulungu and J. Teghem, "Multi-objective Combinatorial Optimization Problems: A Survey", *Journal of Multi-Criteria Decision Analysis*, Vol. 3, 83–104, 1994.

[26] E. L. Ulungu and J. Teghem, "The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems", *Foundations of Computing and Decision Sciences*, 20, pp. 149-165, 1995.

[27] F. Degoutin, X. Gandibleux, "Un retour d'expérience sur la résolution de problèmes combinatoires bi-objectifs", *5e journée du groupe de travail Programmation Mathématique MultiObjectif (PM2O)*, 2002.

[28] F. Glover, M. Laguna, D. De Werra., and E. Taillard, "Tabu search", *Ann. Oper. Res.*, 41. 1992.

[29] G. Mavrotas and D. Diakoulaki, "A branch and bound algorithm for mixed zero-one multiple objective linear programming", *European Journal of Operational Research*, 107 (3), pp. 530-541, 1998.

[30] H. W. Kuhn, "Variants of the Hungarian method for assignment problems", *Operations Research*, vol. 3, pp. 253–258, 1956.

[31] H. W. Kuhn, "The Hungarian method for the assignment problem", *Naval Research Logistics*, 52(1):7–21, 2005.

[32] J. B. Mazzola and A.W. Neebe, Resource-constrained assignment scheduling, *Operations Research*, 34 (4), pp. 560-572, 1986.

[33] K. Fukuda and T. Matsui, "Finding all the perfect matchings in bipartite graphs", *Networks*, 22 (1992), pp. 461-468, 1992.

[34] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network", *Canadian Journal of Mathematics*, vol. 8, pp. 399-404, 1956.

[35] M. Vise, J. Teghem, M. Pirlot and E.L. Ulungu, "Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem", *Journal of Global Optimization* 12, pp. 139-155, 1998.

[36] M. Ehrgott, *Multicriteria Optimization*, 2nd edition, Springer Science & Business Media, 2005.

[37] M. Ehrgott, X. Gandibleux, "Multiobjective combinatorial optimization", In: Ehrgott, M., Gandibleux, X. (Eds.), *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, Kluwers International Series in Operations Research and Management Science, vol. 52, pp. 369-444, 2002.

[38] M. Ehrgott, X. Gandibleux, "Hybrid Metaheuristics for Multi-objective Combinatorial Optimization", *Hybrid Metaheuristics*, pp. 221–259, 2008.

[39] M. Ehrgott and D.M. Ryan, "The method of elastic constraints for multiobjective combinatorial optimization and its application in airline crew scheduling", In: T. Tanino, etc., Editors, *Multi-Objective Programming and Goal-Programming*, pp. 117-122, Springer 2003.

[40] M. Ehrgott, X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization", *OR Spektrum*, 22: 425-460, 2000.

[41] M. M. Zavlanos, L. Spesivtsev, and G. J. Pappas, "A distributed auction algorithm for the assignment problem," In Proc. *47th IEEE Conf. Decision and Control*, Cancun, Mexico, pp. 9–11, 2008.

[42] P. Serafini, "Some considerations about computational complexity for multiobjective combinatorial problems", In *Recent Advances and Historical Development of Vector Optimization*, Lecture notes in Economics and Mathematical Systems vol. 294, 1986.

[43] R. M. Ramos, S. Alonso, J. Sicilia and C. Gonzlez, "The problem of the optimal biobjective spanning tree", *European Journal of Operational Research*, 111, pp. 617-628, 1998.

[44] S. M. Lee, and M. J. Schniederjans, "A multicriteria assignment problem: a goal programming approach", *Interfaces*, 13, (4), 75–81, 1983.

[45] V. Laarhoven, P. J. M. and E. H. L. Aarts, Simulated Annealing: Theory and Applications, Amsterdam: Reidel, 1988.

[46] X. Gandibleux, H. Morita and N. Katoh, "Use of a genetic heritage for solving the assignment problem with two objectives", In Proc. of *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, vol. 2632, pp. 43-57, 2003.