Solution of a Bi-Objective Purchasing Scheduling Problem with Constrained Funds using Pareto Optimization

José Francisco Delgado-Orta[1], Laura Cruz-Reyes[2], Alejandro Palacios-Espinosa[3], and Christian Ayala-Esquivel[1]

[1] Universidad del Mar, Oaxaca, Mexico

[2] Tecnológico Nacional de México, Instituto Tecnológico de Ciudad Madero, Tamaulipas, Mexico

[3] Universidad Autónoma de Baja California Sur, La Paz, Mexico

fdelgado@zicatela.umar.mx, cruzreyeslaura@gmail.com, palacios@uabcs.edu.mx, christiancae@zicatela.umar.mx

Abstract. In this paper the Purchasing Scheduling Problem (PSP) with limited funds is presented. PSP is formulated through the optimization of two objectives based on the inventory-supply process: maximization of satisfied demands and minimization of purchasing costs. The problem is solved using two variants of the Ant Colony System algorithm (ACS), designed under Pareto's optimization principle in which elements of multi-objective representation for computing a feasible solution are incorporated to the basic design of ACS. Experimental results reveal that the Pareto approach improves solutions over the ACS in 8%, obtaining an efficiency of 80% solving the set of PSP instances as purchasing plans. This reveals the advantages of developing evolutionary algorithms based on multi-objective approaches, which can be exploited in planning and scheduling systems.

Keywords: Purchasing scheduling problem, multi objective optimization, ant colony system algorithm

1 Introduction

The purchase of goods is an essential activity for companies and business. It is the process that involves supply based on searches of items in physical facilities, information of products to check inventory stocks, objects or items in big catalogs and supply of goods on supplier locations. All these activities are periodically executed based on customer demands and the inventory control, associated with the availability of economic resources and the storage space in warehouses. In this manner, the Purchasing Scheduling Problem (PSP) (introduced in [1]), establishes a mathematical approach to compute purchasing schedules when demands are variable. Industrial application of PSP is defined as a graph-based problem with several objectives, for example maximization of demand satisfaction, minimization of purchasing costs, maximization of inventory supplies and minimization of supply times.

In addition, multi-objective formulation of PSP faces additional constrains such as penalties to influence a schedule with a subset of desired elements, which implies a quality factors in purchasing related with customer preferences [2, 3], critical supply times [4], negotiations in economical lots of orders [5], categorization of products to be purchased [6], and availability of physical space at warehouse facilities [7] when stock must be supplied. For this reason, selection of appropriated goods to be supplied for inventory has become a complex and multi-objective task, whose approach determines the efficiency of a purchasing plan. It is desirable to optimize economical resources in the companies able to produce, distribute and sell their products according to the supply chain.

2 The Purchasing Scheduling Problem

The Purchasing Scheduling Problem (PSP) is defined through a catalog of products like a weighted graph $G = (V, E)$, where $V = \{P \cup S\}$ consists of a set of n products (P) per m suppliers (S). The set E is formed by pairs (p,s), $p \in P$ and $s \in S$. Each pair has a cost c_{ps} to purchase a product p from any s supplier. Purchasing process is organized through orders $P_k \in P$ (or demands), where k represents a decision maker (a purchaser) with a number n_k of products to be satisfied with an available fund a_k. In these concepts, PSP optimizes two objectives: maximization the amount of satisfied products (for each order P_k) and minimization of purchasing costs (c_{ps}) in an inventory cycle. These objectives introduce the field of multi-objective computation.

3 Multiobjective Optimization

Muitivariant and multiobjective nature of real problems present a challenge to development of efficient algorithms. As a consequence, computation of optimal solutions in a multi-objective problem (MOP) is computationally intractable [8] when large-scale instances are solved. As a consequence, optimal solution of MOP is not possible to compute because MOP is represented by a set of objectives in conflict. This is why computation of solutions in a MOP consists of establish the set Pareto front $PS = \{s_1 , s_2 ,.., s_m\}$ with s_m solution vectors of the problem, where feasibility of solutions is given in terms of dominance and efficiency of Pareto.

Dominance is defined according to the analysis of objectives in pairs. It establishes that objective $s_j \in PS$ dominates a vector $s_j' \in PS$ if and only if $s_j \leq s_j', \forall j \in \{1,..., p\}$, with at least one index j for which the inequality is strict (denoted by $s_j \prec s_j'$). Efficiency of Pareto defines a feasible solution s_j, for which there does is no other solution s_j' such as $z(s_j) \prec z(s_j')$. It implies that s_j is a non-dominated solution (or Pareto optimal). PSP implies the solution of two objectives based on warehouse operations, in which these represent opposite decisions. It defines a multi-objective scene of PSP in terms of a graph-based problem, needed to compute efficient solutions for the related MOP in PSP.

4 PSP Formulation

PSP is formulated through the next data sets:

The general inventory catalogue sets:

P: is the set of products in an inventory catalog with n products.

P_k: is the set of products to be purchased with n_k products, $P_k \in P$, k=1,2,…,s

S: is the set of suppliers in the product catalog with m suppliers.

The model uses the next variables:

k is the number of orders in each inventory cycle. k = 1,2,…, s

c_{ij} is the cost to purchase a product i from a supplier j.

a_k represents the available funds for each order k.

x_{ijk} is an integer variable {0,1}. It has a one value if a product i is assigned to the supplier j in the order k, zero in otherwise.

Objectives of PSP are defined with the f and g coefficients, a normalized objective values in the domain [0,1], where f represents a profit in terms of satisfied demands and g indicates a uniform reference with regard to the assigned c_{ij} values for each assigned product. These values are based on the utility principle proposed in [9;10;11], defined through expressions (1) and (2).

$$\max\ f = \frac{1}{\sum_{k=1}^{s} n_k} \sum_{k=1}^{s} \sum_{i=1}^{n_k} \sum_{j=1}^{m} x_{ijk} \ , \tag{1}$$

$$\min\ g = 1 - \left[\frac{1}{\sum_{i=1}^{n_k} \sum_{j=1}^{m} c_{ij}} \sum_{k=1}^{s} \sum_{i=1}^{n_k} \sum_{j=1}^{m} c_{ij} x_{ijk} \right]. \tag{2}$$

Solution of a multiobjective problem is defined in [12] as a single-objective based on a utility value, following a decomposition strategy. For this reason, objective g is inverted and solved as a maximization objective. As a result, PSP is defined in the general model of expressions (3)-(5):

$$\text{maximize } z = f + g \ , \tag{3}$$

Subject to:

$$\sum_{j=1}^{m}\sum_{i=1}^{n_k} c_{ij} x_{ijk} \le a_k \qquad k = 1,2,...,s \tag{4}$$

$$x_{ijk} \in \{0,1\} \qquad i = 1,2,...,n, j = 1,2,...,m; \ k = 1,2,...,s \tag{5}$$

The z value of equation (3) has a one-value when all products have been assigned (f is optimal and the dominant objective); in the other hand, a zero-value indicates that g is the dominant objective. Expressions (4)–(5) establishes constrains of available funds in the integer model.

5 The Ant Colony System Algorithm

The Ant Colony System (ACS) algorithm [13] is a well-known method to solve graph-based problems. Construction procedures of solutions in ACS are based on selection of arcs (i,j) of a graph. Ants travel around the roads, leaving an amount of pheromone τ_{ij}, used to determine the desirability of the roads η_{ij}. These parameters are used by artificial ants to generate desirable routes, such as the feedback process of natural ants that looks for the shortest paths between the anthill and the food sources. Evolutive process (iterative) of ACS permits evaporation of pheromone trails to converge towards the most feasible routes, which optimize objectives of the problem. General ACS procedure is presented in Fig. 1.

Procedure *ACS_Algorithm* ()
    Initialize_parameters (τ_{ij}, η_{ij})
    While(*isReached*(*stopCriteria*)) do
        constructionProcedure(τ_{ij}, η_{ij})
        updateOfPheromoneTrailsProceduure(τ_{ij})
    End_of_while
End Procedure

Fig. 1. The AntColonySystem Procedure

The *constructionProcedure* in *ACS_Algorithm* builds routes with the desirable nodes in the problem using a transition rule. It defines a basic multi-objective ant colony system algorithm, defined in [1], and based on the multi-objective formulation of PSP. This algorithm creates solutions through of selection of arcs of i products that are purchased to the j suppliers, where selection of the next i-th product is randomly performed in each order P_k. When a product has been selected, the supplier j is chosen using the parameter q_0 of equation (6).

$$j = \begin{cases} \arg\max_{u \in N_i^k} \left\{ [\tau_{ij}]^{\alpha} * [\eta_{ij}]^{\beta} \right\} & q < q_0 \\ f(p_{ij}^k) & \text{otherwise} \end{cases} \tag{6}$$

When $q<q_0$, a deterministic selection is performed using the τ_{ij} and η_{ij} , and the constructive parameters α and β of ant algorithms; otherwise a roulette is executed through the computation of the function f of expression (7). This function is used to explore the neighborhood $N_k(i)$ of suppliers for the i-th product to be selected. This exploration is performed through the p_{ij} values.

$$p_{ij}^k = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha} [\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k} [\tau_{il}]^{\alpha} [\eta_{il}]^{\beta}} & j \in N_i^k \\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

When an ant chooses a feasible arc (i,j), local evaporation of pheromone is performed using the τ_0 values and the $\rho \in [0,1]$ parameter of expression (8). This process is executed while ants have feasible arcs to select in *constructionProcedure.*

$$\tau_0 \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0 \tag{8}$$

Once those ants have completed their solutions in *constructionProcedure*, global updating of pheromone (the *updateOfPheromoneTrailsProceduure*) is performed according to equation (9).

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}\ ,\ \forall (i,j) \in T_k, T_k \subseteq P_k \tag{9}$$

Where $\Delta\tau_{ij}$ represents the amount of deposited pheromone, which is computed like a measure cumulative uniform of the selected products by an ant in an order T_k. Expressions (6)-(9) define the heuristics of the basic design for the ACS algorithm, commonly used in single-objective problems (such as the aggregation described in the PSP formulation). However, solution of PSP requires a diversification of the search in the solution space, needed to reach the best solutions according to the Pareto's efficiency principle. Reason why, the knowledge based on Pareto's approach is incorporated to the ACS design to solve the related multi-objective problem.

6 The Pareto Optimization approach

Pareto Optimization has been used in optimization to obtain a Pareto Optimal Set to solve multiobjective problems [14]. All solutions in the Pareto's set are non-dominated solutions. In this way, Pareto Ant Optimization establishes the set of non-dominated solutions with a number of ant colonies that have the same number of ants. In its evolutive process, solutions of ants are compared and the pheromone updating process is applied to the non-dominated solutions. Pareto Ant Optimization defines two variants over the basic design of ACS to solve multi-objective problems. It is based on multi-objective heuristic rules that permit to guide the ants in the *constructionProcedure* over different regions of the solution space. The first variant (P-ACO^1) defines a modified Δ rule used in global updating of pheromone. It is computed through equation (10).

$$\Delta\tau_{ij} = \frac{1}{n_k}\left(\frac{f+g}{\theta}\right)^2 \tag{10}$$

The Δ values of expression (10) introduce a profit/cost relationship between objectives using a θ value, used according to [15] as a balancing parameter in selection of arcs. Where n_k is the size of the problem. The $\Delta\tau_{ij}$ values in global updating process ensure a faster convergence for ant algorithms in based-graph problems. However, an appropiated θ value can determine a better efficiency in solutions of a MOP.

The second variant (P-ACO^2) consists of introducing pheromone values per each objective τ_{ij}^k. In addition of pheromone values, also heuristic values η_{ij}^k are added according to the *k*–th objective. This strategy permits a further exploration for each single-objective in cases where arcs are selected in non-deterministic way. It is used to determine solutions in the Pareto's front. Consequently, P-ACO^2 algorithm defines the τ_{ij}^f , τ_{ij}^g , η_{ij}^f and η_{ij}^g values, which represents the uniform profit/cost values for objectives in PSP. These parameters define the multi-objective selection rule of expression (11), which defines the third form to compute solutions in the *constructionProcedure* of ACS.

$$p_{ij}^k = \frac{\left[\left(\tau_{ij}^f\right)^{1-\lambda}\left(\tau_{ij}^g\right)^{\lambda}\right]^{\alpha}\left[\left(\eta_{ij}^f\right)^{1-\lambda}\left(\eta_{ij}^g\right)^{\lambda}\right]^{\beta}}{\sum_{l\in N_i^k}\left[\left(\tau_{ij}^f\right)^{1-\lambda}\left(\tau_{ij}^g\right)^{\lambda}\right]\left[\left(\eta_{ij}^f\right)^{1-\lambda}\left(\eta_{ij}^g\right)^{\lambda}\right]} \tag{11}$$

Where $\lambda \in [0,1]$ represents the relative importance of the different objectives according to [16]. Selection of arcs (*i,j*) related to the heuristic rule of equation (11) implies that updating pheromone requires a multi-objective definition in design of the P-ACO^2 variant. Therefore, $\Delta\tau_{ij}^k$ values are introduced as a performance measure of the current solution with regard to the *k* objectives of the problem, used in global upgrading of pheromone. Once that all arcs (*i,j*) are selected in *constructionProcedure*, the P-ACO^2 variant performs an upgrading rule (incorporated in *updateOfPheromoneTrailsProceduure*) that uses the Δ values per each *k* objective ($\Delta\tau_{ij}^f$ and $\Delta\tau_{ij}^g$), defining a dual updating process according to expression (9). Where the Δ values are computed like the inverse of the maximum profit $\Delta\tau_{ij}^f$ and minimum cost $\Delta\tau_{ij}^{g}$ (best ant solutions) respectively. These variants in the multi-objective scene diversify the construction of solutions for PSP, providing to the ACS algorithm different exploration degrees to build feasible solutions for the orders (T_k) according to the formulation of PSP.

7 Architecture of Solution

The proposed approach follows the architecture of Fig. 2. In which the constructive process of purchasing schedules of PSP is described. Architecture consists of two modules: Preprocessing and Optimization. Preprocessing module is used to extract information of PSP sets of a database model (proposed in [17]). This action generates a PSP input instance which consists of a plain-text file, used to establish the solver independent to the database. It permits the use of the architecture in several purchasing scenarios, giving support to the staff of the purchasing personal.

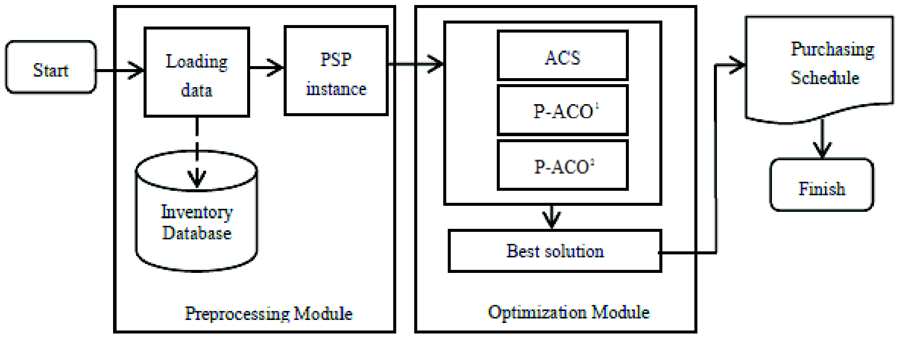

Fig. 2. Proposed solution methology.

Optimization module receives a PSP input instance and executes the basic ACS defined in [1], and the two Pareto-based variants described in this work. In each execution, the best solution of a determined algorithm is presented like a purchasing schedule to decision makers. It represents an optimized solution with regard to the objectives and constrains of PSP, whose efficiency is then analyzed by the purchasers to establish the decision to buy.

8 Experiments and Results

Due to real instances of PSP were unavailable, a dataset of ten orders was built using a pseudo-random number generator. It uses the queries of web catalogs, stored in a model inventory database. The generator creates the orders with different prices and suppliers for products, maximum and minimum prices for products and available funds.

Parameters of generator are shown in Table 1. It supposes an inventory cycle with ten orders (purchasers), where some products have costs more expensive than available funds.

Table 1. Input parameters for instances generated for PSP.

Orders (k)	**Number of Products (n_k)**	**Min price**	**Max price**	**Available Funds**
1	126	14.00	10999.00	25000.00
2	123	73.00	60000.00	50000.00
3	63	29.00	120000.00	40000.00
4	146	99.90	15980.00	65000.00
5	70	3.00	60000.00	30000.00
6	194	56.90	20000.00	80000.00
7	128	75.00	18799.00	75000.00
8	119	95.00	18000.00	55000.00
9	108	3.00	88996.00	48000.00
10	126	14.00	11499.00	40500.00

Instances were solved in an Apple MacBook Pro device model A1286, four-core processor (2.4 Ghz per core), 8 GB of RAM memory, 750 GB hard disk under Mac OS X 10.9 Mavericks. ACS and its multiobjective variants (P-ACO^1 and P-ACO^2) were developed in Java Standard Edition 8 with Eclipse Luna. At each execution, an accumulated sum (∑) is stored with the number of times in which the values of best solutions are reached. It indicates the exploration degree of each algorithm.

Table 2. Solutions of PSP with ACS and ACO variants based on Pareto approach.

Algorithm	**Iterations**	**1000**	**5000**	**10000**	**15000**	**20000**	**30000**
ACS	f	0.8301	0.8513	0.8431	0.8212	0.8205	0.8452
	g	0.1150	0.1005	0.1030	0.1278	0.1288	0.1026
	∑	22.31	24.17	23.32	26.47	25.22	25.53
	t	895.75	8753.96	15830.87	9840.30	18963.43	16732.78
P-ACO^1	f	0.8643	0.8513	0.8216	0.8807	0.8895	0.8772
	g	0.0818	0.0963	0.1230	0.0795	0.0751	0.0894
	∑	16.45	12.29	18.73	17.55	17.34	22.73
	t	667.71	5542.52	8754.93	5503.87	13927.54	29453.01
P-ACO^2	f	0.8801	0.8582	0.8870	0.8925	0.8858	0.8993
	g	0.0849	0.0991	0.0802	0.0755	0.0763	0.0722
	∑	28.72	23.55	30.05	32.92	28.64	30.63
	t	1350.64	6922.03	22532.98	25073.84	21569.23	27954.91

Additionally, average time computation is measured (t) in which best solutions of ACS are reached, and the f and g average values for each instance. Table 2 shows the performance for ACS and Pareto-ACO variants for six tests with 30 executions with: 1000, 5000, 10000, 15000, 20000 and 30000 iterations for each algorithm. Parameter values established to test the algorithms were: $q_0 = 0.5$, ρ=0.1, $\theta = 0.5$, $\lambda = 0.2$,$\alpha = 1$ and $\beta = 2$. In each execution, the best solution is storage to establish the average performance that is presented in Table 2.

Results of Table 2 indicate that ACS reaches a 72% in average of z values, in an average time of 12 seconds. It presents a variation coefficient of 0.76 with a Pearson Correlation Coefficient of -0.98. P-ACO^1 algorithm improves in average 5% the results of ACS, but according to the $\sum$ value, the search is more directed towards a faster convergence ($\sum$ average value is less than ACS). It demonstrates that P-ACO^1 variant over the ACS algorithm is able to improve the solutions of PSP. Even though P-ACO^1 reaches a variation coefficient of 0.81, the Pearson correlation is established in -0.94. It reveals a slow deviation P-ACO^1 directing the search in the Pareto front with regard to ACS. However, results demonstrate P-ACO^1 variant diversifies the search and reaches a faster convergence than ACS.

In the other case, results of the P-ACO^2 algorithm demonstrates that introducing pheromone values per each objective give to the ACS enough exploration degree to improve objective results of ACS by 8%, and 3% in average results of the ACS and P-ACO^1 variants, reaching the 80% in average in z values. Effects of this are shown in the $\sum$ column. This exploration average is supported by a variation coefficient of 0.83 and a correlation coefficient of -0.98, which establishes a diversified and further search in the Pareto Front. Efficiency of P-ACO^2 can be also observed in the average time of computation for the best solutions (17 seconds in average). Although it presents slower convergence, it is proved that P-ACO^2 variant represents the best alternative when testing configurations for the ant algorithms, described in terms of Pareto efficiency solving PSP.

9 Conclusions and Future Works

In this paper, the Purchasing Scheduling Problem was approached with three variants of the Ant Colony System Algorithm. The first variant (ACS) represents an efficient strategy when the search in the problem looks for a single objective, providing good solutions. In the same manner, it was proved that hybridization of the ACS algorithm using the Pareto principles is helpful in discovering different regions of the solution space, giving better solutions with the test parameters.

Consequently, an alternative to compute the Pareto optimal values can be approached using some neighborhood techniques, such as the classical 3-Opt and Cross Exchange operators over the P-ACO^1 and P-ACO^2 algorithms, well-known operators that usually improve results of the ACS algorithm. Additionally, results of P-ACO^1 and P-ACO^2 algorithms to determine the importance of the constructive parameters (θ and λ) in the proposed Pareto's approach.

Efficiency of PSP solutions and speed of computation show the advantages of developing evolutionary algorithms to integrate them in complex decision-making systems. They can be used as planning tools to develop ERP systems (Enterprise Resource Planning), reliable information technology resources to implement in industrial and organizational environments.

References

1. Delgado, O. et al.: An Ant Colony System Metaheuristic for Solving a Bi-Objective Purchasing Scheduling Problem. Research in Computing Sciences 82, pp. 21–30 (2014)
2. Zahra, L., Shahnorbanun, S., Muriati, M.: A Product Quality-Supply Chain Integration Framework. Journal of Applied Sciences 13(1):36–48 (2013)
3. Wallin, C. et al.: What is the “right” inventory management approach for a purchased item? International Journal of Operations and Production Management 26(1):50–68 (2006)
4. Nagurney, A., Yu, M.: Supply Chain Network Design under Profit Maximization and Oligopolistic Competition. Transportation Research 46:281–294 (2010)
5. Hang, Y., Fong, S, Yan, Z.: CSET automated negotiation model for optimal supply chain formation. World Review of Science, Technology and Sustainable Development 7(1):67–78 (2006)
6. Guo, H. et al.: Product Feature Categorization with Multilevel Latent Semantic Association. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 1087–1096 (2009)
7. Hall, N.: The Warehouse-Inventory-Transportation Problem for Supply Chains. Innovative European Journal of Operation Research 237:690–700 (2014)
8. Doerner, K. et al.: Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection. Annals of Operation Research 131(4) (2004)
9. Kumar, R. et al.: Evolution of Hyperheuristics for the Biobjective 0/1 Knapsack Problem by Multiobjective Genetic Programming. In: Conor Ryan & Maarten Keijzer, Ed., GECCO, ACM, pp. 1227–1234 (2006)
10. Falkenauer, E.: A hybrid grouping genetic algorithm for bin packing. Journal of Heuristics, 2:5–30 (1996)
11. Martello, S., Toth, P.: Lower bounds and reduction procedures for the bin packing problem. Discrete Applied Mathematics 28:59–70 (1990)
12. Hamdi, A., Mukheimer, A.: Modified Lagrangian Methods for Separable Optimization Problems. Abstract and Applied Analysis, Vol. 2012, pp. 1–20 (2012)
13. Dorigo, M., Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation, pp. 53–66 (1997)
14. Doerner, K. et al.: Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection. Annals of Operations Research (2004)
15. Chaharsooghi, S., Meimand Kermani, A.H.: An effective ant colony optimization algorithm (ACO) for multi-objective resource allocation problem (MORAP). Applied Mathematics and Computation, vol. 200, pp. 167–177 (2008)
16. Zhao, F., Tang, J., Tang, Y.: A New Approach based on Ant Colony Optimization to determine the Supply Chain Design for a Product Mix. Journal of Computers 7(3):736–743 (2012)
17. Coronel, J.A.: Solution of Assignation problem in purchasing of goods using metaheuristic algorithms. Bachelor’s Thesis, Universidad del Mar campus Puerto Escondido (2014)