AN ITERATIVE PROCEDURE FOR PRODUCTION-INVENTORY-DISTRIBUTION ROUTING PROBLEM

***Dicky Lim Teik Kyee*[1] *and Noor Hasnah Moin*[2*]**

[1,2] Institute of Mathematical Sciences, Faculty of Science, University of Malaya, 50603 Kuala Lumpur, Malaysia.

E-mail: [1]dickylim@siswa.um.edu.my, [2]noor_hasnah@um.edu.my

ABSTRACT

In this paper, the integrated Production, Inventory and Distribution Routing Problem (PIDRP) is modelled as a one-to-many distribution system, in which a single warehouse or production facility is responsible for restocking a geographically dispersed customers whose demands are deterministic and time-varying. The demand can be satisfied from either inventory held at the customer sites or from daily production. A fleet of homogeeous capacitated vehicles for making the deliveries is also considered. Capacity constraints for the inventory are given for each customer and the demand must be fulfilled on time, without delay. The aim of PIDRP is to minimize the overall cost of coordinating the production, inventory and transportation over a finite planning horizon. We propose a MatHeuristic algorithm, an optimization algorithm made by the interpolation of metaheuristics and mathematical programming techniques, to solve the model. In this paper, we propose a two-phase solution approach to the problem. Phase I solves a mixed integer programming model which includes all the constraints in the original model except for the routing constraints. The model is solved by using Concert Technology of CPLEX 12.5 Optimizers with Microsoft Visual C++ 2010. In phase 2, we propose a variable neighborhood search procedure as the metaheuristics for solving the problem. Computational experiment is conducted to test the effectiveness of the algorithm.

Keywords: Production-Inventory-Distribution Routing Problem; MatHeuristics; Mixed Integer Programming; Variable Neighbourhood Search

1.0 INTRODUCTION

In the competitive business environment, many companies face problems with the inventory and distribution management. Thus, they keep searching for ways to design and manufacture new products, and distribute them in an efficient and effective manner. After years of focusing on reduction in production and operation costs, companies are beginning to look into distribution activities as the last component for cost reduction. At the planning level, the goal is to coordinate production, inventory, and delivery to meet customers demand so that the corresponding costs are minimized. Therefore, integrating production and distribution decisions is a challenging problem for manufacturers that are trying to optimize their supply chain. Although the supply chain management literature is extensive, the benefits and challenges of coordinated decision making within supply chain scheduling models have not been studied. In general, the problem of optimally coordinating production, inventory and transportation is called the production-inventory-distribution routing problem (PIDRP) that is know to be NP-hard [1]. The PIDRP is sometimes known as production routing problems [2].

Companies generally need to make decisions on production planning, inventory levels, and transportation in each level of the logistics distribution network in such a way that customer's demand is satisfied at minimum cost. The PIDRP usually arises in the retail industry where customers or outlets rely on a central supplier or manufacturer to provide them with a given commodity on a regular basis. The integration of production, inventory, and distribution increases the complexity of the problem. PIDRP is defined by a combination of a capacitated lot-sizing problem and a capacitated, multi-period vehicle routing problem (VRP). A manufacturer must develop minimum cost production and distribution schedules for a single product that are sufficient to meet all customers demand over the planning horizon. The PIDRP is relevant in Vendor Managed Inventory where the supplier (manufacturer) monitors the inventory at the retailers and coordinate efficient resource utilitization to replenish the retailers.

2.0 LITERATURE REVIEW

The PIDRP is different than the traditional VRPs because it requires multiple customer visits to satisfy demand spread out over an extended period of time. It is most similar to the inventory routing problem, IRP [3-6] and the periodic routing problem, PRP [7-9]. PIDRP can be catergorized based on the underlying assumptions. The number of production plants can be either single or multiple and producing single or multiple products. The inventory policy commonly employed is the maximum or order-up-to level or a combination of both policies. The distribution can be carried out by a fleet of homogeneous or hetrogeneous vehicle with limited capacity. For detail classifications we refer the readers to [2].

There has been a large amount of researches in the areas relating to production, inventory and distribution routing problem. The first paper to discuss PIDRP is due to Chandra [10] and Chandra and Fisher [11] and the authors show the benefit of coordinating the three component which results in 3-20% cost savings compared to sequentially solving the problems separately. Lei et al [12] studied a multi-facility PIDRP with heterogeneous fleet that was motivated by a chemical manufacturer with international customers. The authors proposed a two-phase solution approach where the problem is decomposed into phase 1 in which the model is solved as a mixed integer programming problem subject to all the constraints in the original model except that the transporter routings are restricted to direct shipment between facilities and customer sites. The potential inefficiency of the direct shipment is then solved in Phase 2 where heuristic procedure is applied to solve an associated consolidation problem that is formulated as a capacitated transportation problem with additional constraints. Testing showed that the approach gave good solutions to instances with up to 50 customer sites over 2 to 4 periods.

An alternative approach using Greedy Randomized Adaptative Search Procedure (GRASP) is proposed by Boudia et al. [13]. The PIDRP considered comprises of a single production facility that produces single product. The originality of the approach is to tackle production and routing decisions concurrently instead of resorting to classical two-phase approaches which are still widely used in practice. The two principal difficulties in the GRASP design were to randomize the construction of a trial solution without altering solution quality too much and to develop a local search able to modify both the production plan and the sets of trips on each period. The GRASP considered embeds improvement by a reactive tabu search algorithm for solving the PIDRP. Similarly, Bard and Nananukul [14] developed a reactive tabu search algorithm for solving the PIDRP. An essential component of their methodology was the use of an allocation model in the form of MIP to find good feasible solutions that were used as starting points for the tabu search. The neighborhood consisted of swap and transfer moves. Path relinking was also used in post-processing phase to seek out marginal cost reductions.

Armentano et al. [15] extended the ideas in [13, 14] to include multiple products. The authors presented two heuristic approaches that allow trajectories with feasible and infeasible solutions. The first approach is a tabu search with short memory that uses a compound move at each iteration involving the shift of an amount of an item delivered in a given period to every preceding and succeeding period, the determination of a new route, and the calculation of a new production plan over the time horizon. While the second approach makes use of path relinking that is integrated with tabu search, such that every tabu search local minimum is linked with the farthest solution of a pool of elite solutions. The approaches were tested on a set of small and large generated instances with multiple items. Besides, the approach were also tested on a set of single item instances [13] and they outperformed the memetic algorithm suggested by Boudia and Prins [16] and the reactive tabu search developed by Bard and Nananukul [14].

Due to the complexity of PIDRP, few researchers have proposed exact algorithms. Amongst them is Bard and Nananukul [1] which combined heuristic within the exact branch and price framework. The approach exploits the efficiency of heuristics and the precision of branch and price. The authors devised a new branching strategy to accommodate the unique degeneracy characteristics of the master problem and the algorithms were tested on instances with up to 50 customers and 8 time periods. Adulyassak et al. [17] proposed a branch-and-cut algorithms for both both PIDRP and IRP for the maximum level (ML) and order-up-to level (OU) inventory replenishment policies. The algorithms were tested on IRP and PRP instances with up to 35 customers, three periods, and three vehicles and the authors extended to parallel implementation to be able to solve larger instances. The largest instance for PIDRP on a multicore machine is 35 customers, six periods, and three vehicles.

We refer the readers to the review paper by Sarmento and Nagi [18] and the recent review by Adulyasak et al. [2] which gives a comprehensive state of the art of PIDRP.

The paper is organized as follows. Section 3 reviews the description of PIDRP and its mathematical formulation. In Section 4, the two phase methodolody we proposed to solve the problem is discussed in detail. In Section 5, we present the computational results to evaluate the performance of the algorithm. Finally, we present our conclusions in Section 6.

3.0 PROBLEM DESCRIPTION AND MATHEMATICAL FORMULATION

We consider a production, inventory and distribution routing problem (PIDRP) similar to one proposed by [19]. The problem consists of a single production facility that produces a single product and distributes it to a set of n customers with time varying and non-negative demands d_{it} in each period and can be stored as the inventory by incurring unit holding cost at the production facility as well as the customer sites. If production takes place at the facility in period t, then a setup cost f_t is incurred for $t = 0, 1, \dots, \tau$. In constructing delivery schedules, each customer can be visited at most once per period (split delivery is not allowed) and each of the θ homogeneous vehicles can make at most one trip per period. In this study, the initial inventories at the production facility and customers' sites are assumed to be zero.

Moreover, it is assumed that at the end of planning horizon all inventories (both at the production facility and customers' sites) is required to be zero. It is also assumed that all deliveries takes place at the beginning of the period and arrive at the time to satisfy demand for at least that day. The objective is to construct a production plan and delivery schedule which minimizes production, inventory at the production facility and customers' sites and distribution costs while fulfilling customers' demand. The number of vehicles is given and the total delivery quantity must not exceed vehicle capacity.

The following notations are used in the development of the mathematical formulation.

Indices

i, j	indices for customers, where 0 denotes the depot
t	index for periods
N	set of customers; $N_0 = N \cup \{0\}$ and $\lvert N \rvert = n$
T	set of periods in the planning horizon; $T_0 = T \cup \{0\}$ and $\lvert T \rvert = \tau$

Parameters

d_{it}	demand of customer i in period t
K	number of vehicles
Q	vehicle capacity
c_{ij}	transportation cost from customer i to customer j
C	production capacity
f	fixed production setup cost
h^P	unit production holding cost
I^P_{max}	maximum inventory level at the production facility
h^C_i	unit inventory holding cost at customer site i
$I^C_{max,i}$	maximum inventory level at the customer site, i

Decision Variables

x_{ijt}	1 if customer i immediately precedes customer j on a delivery route in period t; 0 otherwise
y_{it}	load on a vehicle immediately before making a delivery to customer i in period t
w_{it}	amount delivered to customer i in period t
p_t	production quantity in period t
I^P_t	inventory at the production facility at the end of period t
z_t	1 if there is production facility; 0 otherwise
I^C_{it}	inventory at customer site i at the end of period t

The PIDRP can be formulated as follows

$$\emptyset_{IP} = \min \sum_{t \in T} \sum_{i \in N_0} \sum_{j \in N_0} c_{ij} x_{ijt} + \sum_{t \in T} f z_t + \sum_{t \in T_0 \setminus \{\tau\}} h^P I^P_t + \sum_{t \in T \setminus \{\tau\}} \sum_{i \in N} h^C_i I^C_{it} \qquad (1)$$

subjects to:

$$I_t^P = I_{t-1}^P + p_t - \sum_{i \in N} w_{it} \quad ,\forall t \in T_0 \tag{2}$$

$$I_{it}^C = I_{i,t-1}^C + w_{it} - d_{it} \quad ,\forall i \in N, t \in T \tag{3}$$

$$\sum_{i \in N} w_{it} \le I_{t-1}^P \quad ,\forall i \in N, t \in T \tag{4}$$

$$p_t \le C z_t \quad ,\forall t \in T_0 \setminus \{\tau\} \tag{5}$$

$$p_0 \ge \sum_{i \in N} \left(d_{i1} - I_{i0}^C\right) \tag{6}$$

$$\sum_{\substack{j \in N_0 \\ j \ne i}} x_{ijt} \le 1 \quad ,\forall i \in N, t \in T \tag{7}$$

$$\sum_{\substack{i \in N_0 \\ i \ne j}} x_{ijt} = \sum_{\substack{i \in N_0 \\ i \ne j}} x_{jit} \quad ,\forall j \in N, t \in T \tag{8}$$

$$\sum_{j \in N} x_{0jt} \le K \quad ,\forall t \in T \tag{9}$$

$$y_{jt} \le y_{it} - w_{it} + G_t^{\max}\left(1 - x_{ijt}\right) \quad ,\forall i \in N, j \in N_0, t \in T \tag{10}$$

$$w_{it} \le G_{it}^{\max} \sum_{j \in N_0} x_{ijt} \quad ,\forall i \in N, t \in T \tag{11}$$

$$0 \le I_t^P \le I_{\max}^P, \quad 0 \le I_{it}^C \le I_{\max,i}^C; \quad \forall i \in N, t \in T \setminus \{\tau\}; \quad I_\tau^P = I_{i\tau}^C = 0, \quad \forall i \in N \tag{12}$$

$$x_{ijt} \in \{0,1\}, \quad z_t \in \{0,1\}, \quad 0 \le y_{it} \le Q, \quad p_t \ge 0, \quad w_{it} \ge 0 \text{ and integer}, \ \forall i \ne j \in N_0, t \in T \tag{13}$$

where $G_{it}^{\max} = \min\left\{Q, \sum_{l=t}^{\tau} d_{il}\right\}$ and $G_t^{\max} = \min\left\{Q, \sum_{i \in N} \sum_{l=t}^{\tau} d_{il}\right\}$

The objective function (1) expresses the minimization of the sum of transportation costs, production setup costs, holding costs at the production facility and the holding costs at the customer sites. Constraints (2) and (3) represent the equations of inventory flow balance for production facility and customers respectively. The total amount available for delivery on day t is limited by the amount in inventory at the production facility in period $t-1$ as indicated in (4). (5) limits production in period t to the capacity of the production facility, and (6) allows production in period 0. (7) ensures that if customer i is serviced in period t, then it must have a successor on its route, and the route continuity is enforced by (8). (9) limits the number of vehicles that depart form the production facility in period t to the number of vehicles available θ, and (10) keeps track of the load on the vehicles. The value of G_t^{max} is specified to be as small as possible while ensuring that (10) is feasible. The amount delivered to each customer is limited by the parameter G_{it}^{max} in (11). The variable bounds are specified in (12) and (13).

4.0 SOLUTION METHODOLOGY

In this study, we propose a two phase solution approach to solve the PIDRP model. Allocation model which is the simplified version of model above is solved in phase 1 to determine the production capacity, amount delivery to customers and inventories at both production facility and customers' sites. The initial solution to the problem is found in this phase. In phase 2, the delivery routes for each period are constructed based on the customer allocations obtained from phase 1 using giant tour and Dijkstra's algorithm. Next, variable neighborhood search (VNS) is developed to improve the initial solutions.

4.1 Initial Solution

An initial solution can be found in phase 1 by solving the allocation model as a mixed integer programming (MIP) to get a set of feasible allocations. The routing variables, x_{ijt} and the associated constraints (7)-(10) are removed and aggregated vehicle capacity constraint are introduced to the allocation model. The formulation without routing components requires some additional notations: f_{it}^Crepresents the fixed cost of making a delivery to customer i on day t, e_{it}^C represents the variable cost of delivering one item to customer i on day t, and z_{it}^C takes the value 1 if a delivery is made to customer i on day t and 0 otherwise.

As in Nananukul [19], we divide the problem into two cases. For problem instances with $n^2\tau \leq 500$, the routing costs on any period t are approximated by the cost of a round trip between the depot and customer i (round trip value$= 2c_{i0}$), so we use surrogate cost term $\sum_{it} f_{it}^C z_{it}^C$ with $f_{it}^C = 2c_{i0}$ for all i and t. In this instance, e_{it}^C is set to zero. Whereas for the problem instances with $n^2\tau > 500$, we set $z_{it}^C = f_{it}^C = 0$ for all i and t, and the variable cost term $\sum_{it} e_{it}^C w_{it}$ is used for replacement, where e_{it}^C is approximated by the cost of making a delivery to customer i directly from the depot divided by the total demand of customer i in period t (i.e., $e_{it}^C = 2c_{0i}/d$). Since in our instances, $n^2\tau > 500$, we set the variables $z_{it}^C = f_{it}^C = 0$ and the allocation model is simplified as follows.

$$\Phi_{IP} = \min \sum_{t\in T} f_t z_t + \sum_{t\in T}\sum_{i\in N} e_{it}^C w_{it} + \sum_{t\in T_0\setminus\{\tau\}} h^P I_t^P + \sum_{t\in T\setminus\{\tau\}}\sum_{i\in N} h_i^C I_{it}^C \tag{1a}$$

Additional new constraint:

$$\sum_{i\in N} w_{it} \leq 0.8QK \quad ,\forall t\in T \tag{14}$$

The primary difference between models (1) and (1a) is that the routing variables (x_{ijt}) and related constraints have been removed. (14) limit the total amount that can be delivered in period t to a fixed percentage of the total transportation capacity, and provides a hedge against the need for split deliveries. The authors in [19] showed that a value of 80% always yielded feasible solutions.

4.2 Variable Neighborhood Search

Variable neighborhood search was initially proposed by Mladenovic and Hansen [20] for solving combinatorial and global optimization problems. The main reasoning of this metaheuristic is based on the idea of a systematic change of neighborhoods within a local search method. Exploration of the search space is carried out by the local search which allows the algorithm to jump from one neighborhood to another. This allows the algorithm to escape from the local optimum.

Let us denote a finite set of pre-selected neighborhood structures with $\mathcal{N}_k$, where $k = 1, \ldots, k_{max}$, where k_{max} refers to the maximum number of neighborhood used, and $\mathcal{N}_k(x)$ the set of solutions in the kth neighborhood of x. The stopping condition may be maximum number of iterations, maximum number of iterations between two improvements or maximum CPU time allowed. There are three phases of the main VNS: *Shaking*, *Local Search*, and *Move or Not*. The basic VNS heuristic comprises the steps given in Fig. 1. The flow chart of the VNS is outlined in Fig. 2.

```
Initialization. Step 0: Define a set of neighborhood structures $\mathcal{N}_k$, $k = 1, \ldots, k_{max}$, that will be used in the
                        search and a set of local searches $\mathcal{R}_l$, $l = 1, \ldots, l_{max}$; generate an initial solution $x$;
                        choose a stopping condition;

Repeat the following steps until the stopping condition is met:
        Step 1: Set $k \leftarrow 1$;
        Step 2: Until $k = k_{max}$, repeat the following steps:
                (a) Shaking. Generate a point $x'$ at random from the $k^{th}$ neighborhood of $x$ ($x' \in N_k(x)$)
                (b) Local Search. Apply some local search method with $x'$ as initial solution;
                        denote with $x''$ the so obtained local optimum;
                (c) Move or not. If this local optimum is better than the incumbent, move there ($x \leftarrow x''$),
                        and go back to (1); otherwise, set $k \leftarrow k + 1$.
```

Fig. 1. Steps of basic VNS

Neighborhood Structures

There are several ways that can be used to define the neighborhood structure, for example: 1-interchange (or vertex substitution), symmetric difference between two solutions, Hamming distance, vertex deletion or addition, node based or path based and k-edge exchange. As PIDRP is almost similar to IRP, it comprises of three important components; the production, inventory and routing. However the tradeoff between inventory holding cost at the customers site and the traveling costs. The choice of a suitable neighborhood structure in an PIDRP is not straight forward. The neighborhood can be defined as the symmetric difference between the different clusters within the same period or the symmetric difference between the numbers of customers visited in each period.

In this study our neighborhood comprises of four type of neighborhood structure: as a distance function, that is the cardinality of the symmetric difference between any two solutions V_1 and V_2 written as $\rho(V_1, V_2) = |V_1 \backslash V_2|$ using the forward and backward transfer, the swap and the tranfer (insertion). The individual neighborhood structure is described in details in Section 4.2.1.

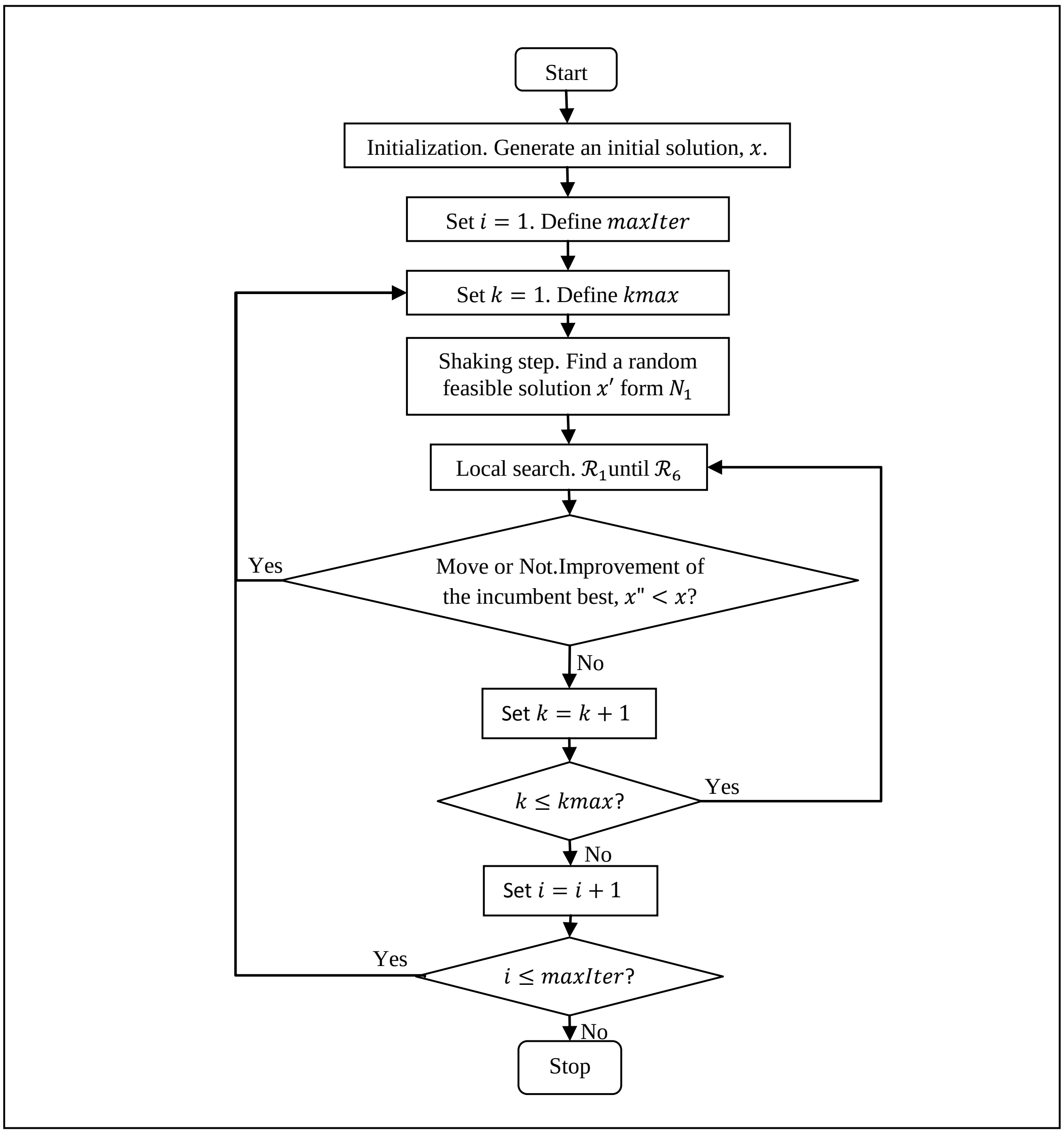
Start
Initialization. Generate an initial solution, x.
Set $i = 1$. Define $maxIter$
Set $k = 1$. Define $kmax$
Shaking step. Find a random feasible solution x' form N_1
Local search. $\mathcal{R}_1$until $\mathcal{R}_6$
Move or Not.Improvement of the incumbent best, $x" < x$?
Yes
No
Set $k = k + 1$
$k \leq kmax$?
Yes
No
Set $i = i + 1$
$i \leq maxIter$?
Yes
No
Stop

Fig 2. Flow chart of VNS

4.2.1 Step 0: Initialization

The initial solution is obtained in two steps; (a) construct a giant tour using the sweep algorithm as descibed in [21] and (b) find the corresponding optimal fleet size by constructing cost network and subsequently applying Dijkstra's algorithm which provides an initial feasible solution that contains routes. First, a giant tour is constructed that includes all the customers obtained in phase 1. We define a tour $G = (V, A)$ with $V = \{v_1, v_2, \dots, v_n\}$, the set of nodes representing all the customers' positions in the tour, and $A = \{(v_i, v_j): v_i, v_j \in V\}$ form the arcs which maintain the order of the customers, together with a distance cost c_{ij}. Define a path starting from the depot to the closest customer, and this step is repeated at each node $v_i \in V$ where $i = 1, 2, 3, \dots, n$ with n denoting the total number of customers to be served in the current period, until the last customer is reached. In order to apply Dijkstra's algorithm, we first construct a cost network considering customer data, capacity constraint, distance constraint, and vehicle unit variable and fixed costs. For illustration, consider 12 customers making up the following giant tour $\sigma = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$ with customer demand $d = (4, 2, 2, 2, 1, 4, 2, 2, 2, 1, 4, 2)$. Assume that there is only one type of vehicle, with maximum capacity of 10 units. Let c_{ij} be the distance between node i and node j.

We start to construct this cost network by calculating the cost from the depot, denoted by 0, to customer 1 and from this customer to the depot (return journey) as the cost of arcs $0 - 1$. This is express as $C_{01} = F + v(2d_{01})$. If the total demand of both customers 1 and 2 does not violate the capacity constraint of the vehicle, we calculate the cost of the arcs $1 - 2$ as $C_{02} = F + v(d_{01} + d_{12} + d_{20})$. We continue with this cost construction until the vehicle is full, and then we start using the next vehicle. Fig. 3 shows that we can only have customers $1, 2, 3$ and 4 can then be visited by the vehicle. The process is continued until there is no more arcs connecting the last customer in the giant tour. In general, the cost of arc ij is defined as in $Eq. (15)$.

$$C_{ij} = F + v\left(d_{0,i+1} + \sum_{k=i+1}^{j-1} d_{k,k+1} + d_{j,0}\right) \qquad (15)$$

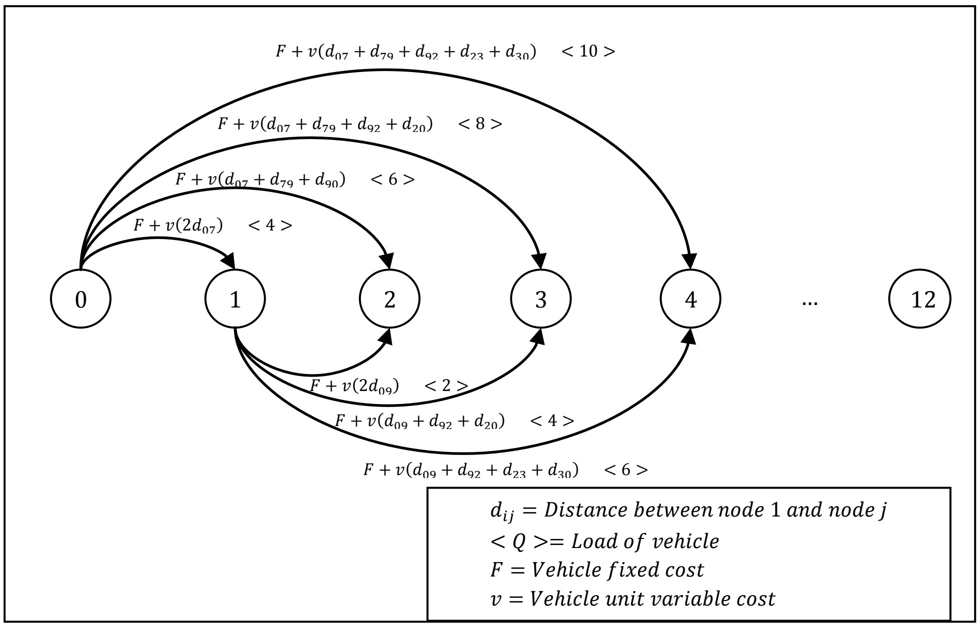
$F + v(d_{07} + d_{79} + d_{92} + d_{23} + d_{30})$ $< 10 >$
$F + v(d_{07} + d_{79} + d_{92} + d_{20})$ $< 8 >$
$F + v(d_{07} + d_{79} + d_{90})$ $< 6 >$
$F + v(2d_{07})$ $< 4 >$
0
1
2
3
4
...
12
$F + v(2d_{09})$ $< 2 >$
$F + v(d_{09} + d_{92} + d_{20})$ $< 4 >$
$F + v(d_{09} + d_{92} + d_{23} + d_{30})$ $< 6 >$
$d_{ij} = Distance\ between\ node\ 1\ and\ node\ j$
$< Q >= Load\ of\ vehicle$
$F = Vehicle\ fixed\ cost$
$v = Vehicle\ unit\ variable\ cost$

Fig. 3. Construction of cost network

After creating the cost network, whose origin is depot $'0'$ and the destination is the last node in the giant tour, Dijkstra's algorithm is applied to obtain the initial feasible solution. This procedure is repeated for each period considered. After an initial feasible solution is found, set $x_{best} \leftarrow x$ and proceed to Step 1.

Defining the stopping condition can differ from one program to another. Most algorithms adopt the maximum number of iterations as tje stopping condition. Other criteria such as maximum running time or cpu time allowed, or number of iterations between two improvements can also be defined in the algorithm.

4.2.2 Step 2(a): Shaking

In this step, a solution x' is picked randomly from the k^{th} neighborhood of the current solution, x. This will ensure that the solution is not far from the current best solution x. We consider four moves, forward transfer, backward transfer, swap and transfer for VNS. The steps of the forward and backward transfer in the shaking step are as follows. The algorithm of the shaking step is shown in Fig. 4.

```
Set num=1
While (num<=k) do    //the number of changes depend on the value of k
{
        Randomly generate the value of r where 0<r<1. Define p1, p2 and p3
                if (r<=p1)      // the value of p1 represent the chosen probability
                    Apply forward transfer
                elseif (p1<r<=p2)
                    Apply backward transfer
                elseif(p2<r<=p3)
                    Apply swap
                else
                    Apply transfer
                endif
                num=num+1
}
```

Fig. 4. The algorithm of shaking step

4.2.2.1 Neighborhood Structure

We consider four neighborhood structures for each $\mathcal{N}_k$: forward and backward transfers, swap and transfer. The aim of the forward transfer is to reduce the inventory holding cost without increasing drastically the transportation cost. In the backward transfer the preference is given to the suppliers with the lower holding cost in order to determine whether the transportation and the inventory holding cost can be futher consolidated. Examples of the forward and backward transfers are illustrated in Figure 1 and 2, respectively. In these example we assume that the coordinate of the 5 customers are $c_1(-2,0), c_2(1,1), c_3(4,2), c_4(3,5), c_5(1,-2)$ and the depot is located at $D(0,0)$ the holding cost per unit for each customer are $h_1 = 12, h_2 = 9, h_3 = 6, h_4 = 3$and $h_5 = 6$ and the vehicle capacity is 10. Note that the routing are separated by zeros and w_{it} and I_{it} are the pick-up quantity and the inventory respectively.

Forward Transfer

Fig. 5 illustrates a forward transfer and the selection of period and supplier to be transferred is biased towards customers with high holding cost. In this example, we select customer in the period 1. Note that we limit the transfer to at most 2 periods only. This is to ensure that the increase in the routing cost is not exceedingly high.

The demand for customer 1 in period 1, 2, and 3 are $d_{11} = 4, d_{12} = 2$ and $d_{13} = 4$. From the figure $I_{11} = 6$ and $I_{12} = 4$, the resultant holding cost for periods 1, 2 and 3 are 81, 24 and 36 respectively, and the total cost, including the routing cost for all 3 periods is 240.3398. Customer 1 is not visited in the period 2 and 3, so we apply forward transfer by inserting customer 1 to period 3 according to the best insertion. Note that inserting customer 1 in period 2 results in the violation of vehicle capacity constraint. The saving after the transfer is 240.3398-153.4129= 86.9269.

Before	Period	Route									Route Cost	Holding Cost
	1	0	2	5	0	1	0	3	4	0	24.1156	81
	w_{it}		2	3		10		2	5			
	I_{it}		0	1		6		0	1			
	2	0	2	3	0						9.0486	72
	w_{it}		2	5								
	I_{it}		0	4								
	3	0	2	0	5	4	0				18.1756	36
	w_{it}		6		2	4						
	I_{it}		4		0	0						
Total Cost											51.3398	189
After	**Period**	**Route**									**Route Cost**	**Holding Cost**
	1	0	2	5	0	1	0	3	4	0	24.1156	33
	w_{it}		2	3		6		2	5			
	I_{it}		0	1		2		0	1			
	2	0	2	3	0						9.0486	24
	w_{it}		2	5								
	I_{it}		0	4								
	3	0	2	0	1	5	4	0			21.5450	36
	w_{it}		6		4	2	4					
	I_{it}		4		0	0	0					
Total Cost											60.4129	93

Fig.5. Example of Forward Transfer

Backward Transfer

The selection of period and customer to be transferred is favorable toward the lower holding cost in the backward tranfer shown in Fig. 6. In this example, we select customer 4 in the period 5. The saving is found by increasing the inventory cost and decrease in routing.

Initial routing for period 4 and 5, with the route cost 18.771591 and 24.36395 and 0 holding cost. According to the inventory updating mechanism, we transfer customer 4 in the period 5 to period 4. As the same customer is visited in period 4, so we embed it together, in which we note that the resulting transfer does not violate the capacity constraint. After the transfer of delivery amount by 2 units, we have a holding cost 6 with $h_4 = 3$. Customer 4 will be eliminated in period 5. The overall savings after the transfer is 64.2431 -58.5812= 5.6619.

Before Transfer	Period	Route							Route Cost	Holding Cost
	4	0	1	5	0	4	0		19.5035	18
	w_{it}		4	4		**4**				
	I_{it}		0	0		**0**				
	5	0	1	5	3	0	4	0	26.7396	0
	w_{it}		4	4	2		2			
	I_{it}		0	0	0		0			
Total Cost									46.2431	18
After Transfer	Period	Route							Route Cost	Holding Cost
	4	0	1	5	0	4	0		19.5035	24
	w_{it}		4	4		**6**				
	I_{it}		0	0		**2**				
	5	0	1	5	3	0			15.0777	0
	w_{it}		4	4	2					
	I_{it}		0	0	0					
Total Cost									34.5812	24

Fig.6. Example of Backward Transfer

Swap

The swap involves an exchange of delivery quantities between two customers i_1 in period t_1 with quantity $\overline{w}_{i_1t_1}$ and i_2 in period t_2 with quantity $\overline{w}_{i_2t_2}$, where t_2 is the first period after t_1 such that $\overline{w}_{i_2t_2} > 0$. For customer i_1, the move considers the maximum portion of $\overline{w}_{i_1t_1}$ that can be reassigned to period t_2 without causing a shortage in period t_1 to be exchanged with full amount $\overline{w}_{i_2t_2}$. If customer i_1 was not scheduled for a delivery in period t_2, then he must be inserted into one of the θ routes. In general, a swap produces a change in holding costs and a change in holding costs in period t_1 and t_2.

Transfer

Similar to backward transfer but we limit the number of periods to be inserted to at least two preceding periods. (i.e. $t_1 - t_2 \geq 2$). The transfer examines each customer i one at a time and tries to reassign the delivery quantity $\overline{w}_{it_1}$ scheduled for t_1 to the latest period, call it t_2, preceding t_1 in which a delivery is scheduled for at least one customer i; that is, $t_2 = \max\{t: t < t_1, \exists \overline{w}_{it} > 0 \text{ for some } i \in N\}$.

We also incorporate the concept of tabu search which forbid the movement of the customer for a few iterations if the customer is chosen to transfer or swap. In all the four moves, only moves that result in feasible solutions are allowed so it is necessary to check for violations of the production constraints and the inventory bounds at the plant and the customer sites, as well as the vehicle capacity constraint.

4.2.3 Step 2(b): Local Search

In our study, the local search consists of six refinement procedures adopted from Imran and Salhi [22]. The order of the refinement procedure is as follows: the 1-insertion inter-route as $\mathcal{R}_1$, the 2-opt inter-route as $\mathcal{R}_2$, the 2-opt intra route as $\mathcal{R}_3$, the swap intra route as $\mathcal{R}_4$, 1-insertion intra-route as $\mathcal{R}_5$, and at last the 2-insertion intra-route as $\mathcal{R}_6$. The process starts by generating a random feasible solution x' from N_1, which is used as temporary solution. The multi-level approach then starts by finding the best solution x'' using $\mathcal{R}_1$. If x'' is better than x', then $x' = x''$ and the search return to $\mathcal{R}_1$, otherwise the next refinement procedure, $\mathcal{R}_2$ is applied. This process is repeated until $\mathcal{R}_6$ cannot produce a better solution.

4.2.4 Step 2(c): Move or Not

If the solution obtained by the multi-level approach, x'', is better than the incumbent best solution x, then set $x = x''$ and the search returns to N_1. But if x'' is found to be worse or same as x, we generate x' from the next neighborhood say $N_k(x)$ and go back to step (2b) again. The process is repeated until the search reaches $N_{k_{max}}$.

5.0 COMPUTATIONAL RESULTS

All the algorithms are written in Microsoft Visual C++ 2010 and performed on 3.1 GHz processor with 8GB of RAM. The code for the allocation model were implemented as mixed integer programming in Concert technology of Microsoft Visual Studio 2010 linked to the CPLEX 12.5 libraries. CPU times were obtained using the time function in C++.

In this paper, we used a data set provided by Boudia et al. [13] consisting of 30 instances of 50 customer problem with a 20-period planning horizon and holding cost $h^P = 1$, $h_i^C = 0$ for all $i \in N$. These instances were randomly generated on a 100×100 Euclidean grid. For each customer i, demand was uniformly distributed between 0 and the storage capacity $I_{max,i}^C$. The vehicle capacities, $Q_{50} = 8000$ and the number of vehicles, $\theta_{50} = 5$.

Our tests were compared to the GRASP [13] and the Memetic Algorithm with Population Management (MA|PM) [16]. Column 2 and 3 give the best solutions for both GRASP and MA|PM. The last six columns illustrate results for our 2-phase methodology tabulating the best objective function, mean and standard deviation over the 10 runs. The last three columns display the computational time, its mean and standard deviation. We have improved 13 out of 30 solutions as compared to Memetic Algorithm with Population Management and our results are superior on all instances when compared to GRASP.

Table 1. Results for 50 customer-20 period

Inst.	GRASP	MA|PM	Our Algorithm					
	Total Cost	Total Cost	Best Cost	Mean	Stand. Dev.	Time (s)	Mean	Stand. Dev.
1	440505	378378	404597	410146.8	2946.97	280.06	278.54	20.77
2	448695	403913	401127	404570.6	1786.47	280.93	289.07	16.49
3	419730	409573	400791	405490.1	2892.42	272.31	290.23	18.19
4	456398	399220	403574	407193.3	2867.57	318.56	299.18	14.43
5	434466	422279	410873	414941.3	2245.13	334.20	308.56	19.10
6	452564	407122	405087	410837.7	2810.77	264.38	290.72	22.01
7	436812	414977	415684	419104.6	1972.51	383.79	360.96	18.78
8	420935	379744	406108	409896.7	2489.10	300.89	323.40	23.31
9	434789	407935	400572	403844.4	2097.15	266.50	274.85	10.55
10	436221	396258	400522	402856.3	1487.82	264.99	273.42	28.89
11	433890	402475	393563	397689.1	2326.93	253.19	307.12	26.52
12	452705	358702	395480	398464.3	2146.70	291.24	272.32	19.75
13	440771	371030	391643	395742.1	2995.11	247.49	249.72	10.91
14	419412	406114	396787	400078.5	1734.72	265.80	278.29	23.32
15	453875	373076	425952	430033.9	2023.24	395.34	386.90	25.28
16	457310	379404	398141	402162.3	2326.66	323.48	320.12	19.13
17	455663	406353	410069	413594.3	2052.57	314.83	287.58	26.08
18	441685	401179	399072	403830.2	2984.88	319.18	328.93	19.77
19	418896	406893	395170	398922.1	2125.64	231.14	258.18	19.24
20	452183	398508	402284	405264.4	2050.40	248.47	282.96	22.99
21	409677	397112	399349	404670.2	2366.29	295.42	298.62	14.42
22	429116	358749	397730	401398.3	1902.57	261.51	280.58	14.87
23	443184	407369	398835	404072.1	2601.56	269.61	266.60	12.08
24	426113	369784	397148	400217.7	1600.80	265.54	290.78	12.61
25	462245	411556	399765	403415.6	2192.09	274.23	265.09	20.54
26	442029	408704	407799	412178.4	1984.58	288.01	294.74	22.83
27	444695	366197	391664	395134.0	2379.64	249.91	285.31	15.94
28	449894	401032	396648	399283.9	1816.52	301.78	280.60	16.68
29	461555	384282	408039	412062.1	2760.34	274.71	276.22	22.31
30	434006	369959	405064	409614.4	2343.33	269.12	282.85	24.19

6.0 CONCLUSION

In this paper, we propose a two phase methodology to solve the production-inventory-distribution routing problem (PIDRP). The problem is decomposed into two parts, allocation model to determine the amount to deliver and the inventory and routing algorithm. The problem comprises of a single product, multi-period in a finite planning horizon. Phase 1 solves the mixed integer programming allocation model and routes are constructed using a giant tour procedure in phase 2 to form a feasible solution. The solution is then improved using the well known algorithm variable neighbourhood search. Testing on benchmark instances show that our proposed algorithm can obtain high quality solutions in a reasonable computational time.

ACKNOWLEDGEMENT

This research received support from University of Malaya Research Grant UMRG RG116/10AFR and the first author would like to acknowlegdge the support from MyMaster programme under Department of Higher Education, Ministry of Education.

REFERENCES

[1] J. F. Bard, and N. Nananukul, "A branch-and-price algorithm for an integrated production and inventory routing problem". *Computer & Operations Research*, vol. 37, no. 12, 2010, pp. 2202-2217.

[2] Y. Adulyasak, J.-F. Cordeau, R. Jans, "The production routing problem: A review of formulations and solution algorithms". *Computers & Operations Research*, vol. 55, March 2015, pp. 141-152.

[3] T. F. Abdelmaguid, and M. M. Dessouky, "A genetic algorithm approach to the integrated inventory distribution problem". *International Journal of Production Research*, vol. 44, no. 21, 2006, pp. 4445-4464.

[4] J. F. Bard, L. Huang, P. Jaillet, and M. Dror, "A decomposition approach to the inventory routing problem with satellite facilities". *Transportation Science*, Vol. 32, No. 2, 1998, pp. 189-203.

[5] B. Golden, A. Assad, and R. Dahl, "Analysis of a large scale vehicle routing problem with an inventory component". *Large Scale Systems*, Vol. 7, No.2-3, 1984, pp. 181-190.

[6] M. Dror, and M. Ball, "Inventory/routing: reduction from an annual to a short period problem". *Naval Research Logistics Quarterly*, Vol 34, No. 4, 1987, pp. 891-905.

[7] M. Gaudioso, and G. Paletta, "A heuristic for the periodic vehicle routing problem". *Transportation Science*, Vol. 26, No. 2, 1992, pp. 86-92.

[8] M. Mourgaya, and F. Vanderbeck, "Column generation based heuristic for tactical planning in multi-period vehicle routing". *European Journal of Operational Research*, Vol. 183, No. 3, 2007, pp. 1028-1041.

[9] P. Parthanadee, and R. Logendran, "Periodic product distribution from multi-depots under limited supplies". *IIE Transactions on Scheduling & Logistics*, Vol. 38, No. 11, 2006, pp. 1009-1026.

[10] P. Chandra, "A dynamic distribution model with warehouse and customer replenishment requirements". *Journal of the Operational Research Society*, Vol. 44, No. 7, 1993, pp. 681-692.

[11] P. Chandra, M. Fisher, "Coordination of production and distribution planning". European Journal of Operational Reserch, Vol. 72, No. 3, 1994, pp. 503-517.

[12] L. Lei, S. Liu, A. Ruszczynski, and S. Park, "On the integrated production, inventory and distribution routing problem". *IIE Transactions on Scheduling & Logistics*, Vol. 38, No. 11, 2006, pp. 955-970.

[13] M. Boudia, M. A. O. Louly, and C. Prins, "A reactive GRASP and path relinking for a combined production-distribution problem". *Computers & Operations Research*, Vol. 34, No. 11, 2007, pp. 3402-3419.

[14] J. F. Bard, and N. Nananukul, "The integrated production-inventory-distribution routing problem for a single commodity". *Journal of Scheduling*, Vol. 12, No. 3, 2009, pp. 257-280.

[15] V. A. Armentano, A. L. Shiguemoto, and A. Lokketagen, "Tabu search with path-relinking for an integrated production-distribution problem". *Computers & Operations Research*, Vol. 38, No. 8, 2011, pp. 1199-1209.

[16] M. Boudia, C. Prins, "A memetic algorithm with dynamic population management for an integrated production-distribution problem". *European Journal of Operational Research*, Vol. 195, 2009, pp. 703-715.

[17] Y. Adulyasak, J.-F. Cordeau, R. Jans, "Formulations and branch-and-cut algorithms for multi-vehicle production and inventory routing problems". *INFORMS Journal on Computing*, Vol. 26, No. 1, 2013, pp. 103-120.

[18] A.M. Sarmiento, R. Nagi, "A review of integrated analysis of production-distribution systems". *IIE Transactions*, Vol. 31, No. 11, 1999, pp. 1061-1074.

[19] N. Nananukul, "Lot-sizing and inventory routing for a production-distribution supply chain". PhD dissertion, Graduate Program in Operations Research and Industrial Engineering, The University of Texas, Austin, 2008.

[20] N. Mladenovic, and P. Hansen, “Variable neighborhood search”. *Computers and Operations Research,* Vol. 24, 1997, pp. 1097-1100.

[21] B. E. Gillett, and L. R. Miller, A heuristic algorithm for the vehicle dispatch problem. *Operations Research*, Vol. 22, 1974, pp. 340-344.

[22] A. Imran, S. Salhi, and N. A. Wassan, “A variable neighborhood-based heuristic for heterogeneous fleet vehicle routing problem”. *European Journal of Operational Research*, Vol. 197, No. 2, 2009, pp. 509-518.