Provided for non-commercial research and education use.
Not for reproduction, distribution or commercial use.

Volume 211, Issue 1, 16 May 2011
ISSN 0377-2217
ELSEVIER
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
Available online at www.sciencedirect.com
ScienceDirect
www.elsevier.com/locate/ejor

This article appeared in a journal published by Elsevier. The attached copy is furnished to the author for internal non-commercial research and education use, including for instruction at the authors institution and sharing with colleagues.

Other uses, including reproduction and distribution, or selling or licensing copies, or posting to personal, institutional or third party websites are prohibited.

In most cases authors are permitted to post their version of the article (e.g. in Word or Tex form) to their personal website or institutional repository. Authors requiring further information regarding Elsevier's archiving and manuscript policies are encouraged to visit:

http://www.elsevier.com/copyright

Author's personal copy

European Journal of Operational Research 211 (2011) 47–56

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

Modeling of financial supply chain

Sushil Gupta [*], Kaushik Dutta [1]

College of Business Administration, Florida International University, 11200 SW 8th Street, Miami, FL 33199, United States

ARTICLE INFO

Article history:
Received 2 November 2009
Accepted 3 November 2010
Available online 11 November 2010

Keywords:
Scheduling
Supply-chain
Accounts payable
Accounts receivables
Cash management

ABSTRACT

The research work on supply-chain management has primarily focused on the study of materials flow and very little work has been done on the study of upstream flow of money. In this paper we study the flow of money in a supply chain from the viewpoint of a supply chain partner who receives money from the downstream partners and makes payments to the upstream partners. The objective is to schedule all payments within the constraints of the receipt of the money. A penalty is to be paid if payments are not made within the specified time. Any unused money in a given period can be invested to earn an interest. The problem is computationally complex and non-intuitive because of its dynamic nature. The incoming and outgoing monetary flows never stop and are sometimes unpredictable. For tractability purposes we first develop an integer programming model to represent the static problem, where monetary in-flows and out-flows are known before hand. We demonstrate that even the static problem is NP-Complete. First we develop a heuristic to solve this static problem. Next, the insights derived from the static problem analysis are used to develop two heuristics to solve the various level of dynamism of the problem. The performances of all these heuristics are measured and presented.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Supply-chain management revolves around coordination and cooperation among several business partners that are linked through flows of material, money and information. These partners include suppliers of basic raw materials and component parts, manufacturers, wholesalers, distributors, transporters, retailers, banks and financial institutions. In general the materials, component parts and finished goods flow downstream although the returned merchandise flows upstream. The money flows upstream in a supply chain whereas the information flows in both directions (Fig. 1). For an effective supply chain system, the management of upstream flow of money is as important as the management of downstream flow of goods.

The problem of flow of goods in supply chains has been studied widely (Kouvelis et al., 2006). Research on supply chain systems has focused on inventory cost, transportation cost and cost related to goods procurement. However, there has been very little research work that focuses on the upstream flow of money.

There are several fundamental differences between the downstream flow of goods and materials and upstream flow of money. In downstream flow, holding of goods and materials increases the inventory holding cost whereas in upstream flow of money, holding of money earns interest which is a completely opposite situation to that of downstream flow of goods. Further, the amount of goods and materials to be delivered downstream depends on the orders placed from the downstream partner and remains constant if the order size is not changed. However, in upstream flow of money, the amount to be paid to an upstream partner will depend on the terms of payment that may include penalty for late payments and/or discounts for early payments. Trade credit policies are well researched in the finance literature, for example, see Borde and McCarty (1998). These differences make the management of upstream flow of money an important and distinct research problem.

The motivation for studying the problem proposed in this paper emerged through our discussions with Sukrit Agrawal (2008), CEO, American Medical Depot, a distributor of medical and surgical supplies in Miami, Florida, USA. According to Agrawal, his company at any time has close to 500 invoices to be paid; and almost the same number of accounts receivables. An optimization model to schedule payments can definitely benefit the company.

This research problem of managing the flow of money is computationally complex for several reasons. First, the inflows and outflows of cash are continuous throughout an organization‘s life span. These flows never cease to exist and, therefore, the problem is dynamic in nature. Second, future cash inflows and outflows are mostly unknown, because such inflows and outflows depend on movement of goods which again depends on market demand. Third, even if the future values of such inflows and outflows are known before hand the problem is computationally NP-Complete. We demonstrated this by first developing a static version of the

* Corresponding author. Tel.: +1 305 348 3248.
E-mail addresses: guptask@fiu.edu (S. Gupta), kaushik.dutta@fiu.edu (K. Dutta).
[1] Tel.: +1 305 348 3248.

0377-2217/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2010.11.005

Author's personal copy

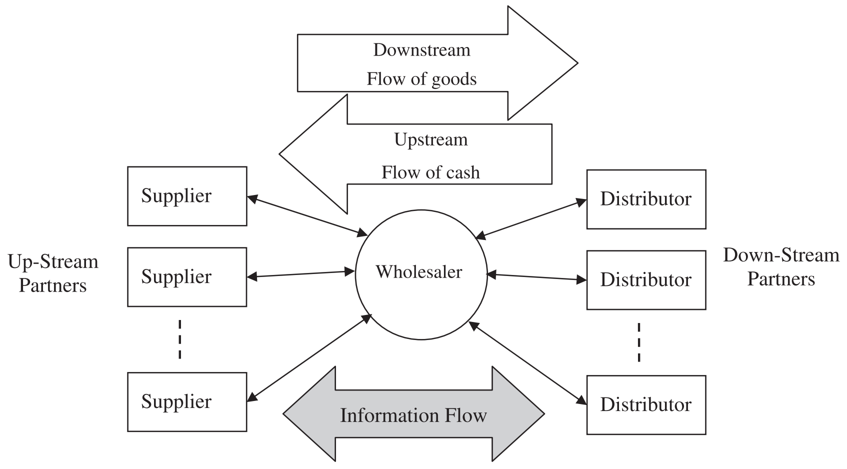

Fig. 1. Supply chain of goods and cash for a wholesaler.

problem. So the problem requires focused study and development of appropriate quality heuristic, which is the main objective of this paper.

To illustrate the complexity of the problem, let us consider a very simple case of a wholesaler, who receives \$100 each day from his distributors; and has to pay two invoices in the amounts of \$1000 and \$2000 from two of his suppliers respectively. Let us assume that the deadline for paying the \$1000 to supplier A is within the next 10 days; and the deadline for paying the supplier B is within the next 15 days. Penalties are to be paid if the payments are made after the deadlines. Suppose the penalties per day are 1.0% and 1.5%, respectively, for the two suppliers. Obviously, payments to the two suppliers cannot be made within the deadlines and the wholesaler needs to decide the order in which the two suppliers will be paid. If the wholesaler decides to pay supplier A first and then supplier B, then the total cash outflow will be \$3859 by 39th day. In this decision the payment to supplier A is made on the 10th day without a penalty; and the payment to supplier B is made on 39th day with penalty. If the wholesaler decides to pay supplier B first and then supplier A, then the total cash outflow is \$3549 by 36th day. In this decision, the payment to supplier B is made on 23rd day and the payment to supplier A is made on 36th day. This decision involves paying penalties to both suppliers but entails a smaller cash outflow; and is the preferred choice. This is a very non-intuitive solution even for such a small scenario. Deriving the optimal scenario for this simple case requires solving multiple non-linear equations. Real life situations are much more complex; and involve multiple suppliers and multiple distributors. The terms and conditions for making payments to various suppliers vary from each other. The real problem is also not static because new invoices continue to arrive while the old invoices still need to be paid. The cash inflow rate may not be continuous and uniform each day. The wholesaler may use rudimentary heuristics such as “first come first served”, or “least penalty”, which may only result in suboptimal decisions. Finding optimal solution to such problem is computationally complex.

In this paper first we develop an integer programming model for the static case where the future receipts from distributors, and the payment terms and amounts of all suppliers are known. We prove that the problem is NP-Complete. Next, we present a heuristic approach where the cash in-flows from distributors are known but the amounts and payment terms from suppliers are unknown. Lastly, we present a dynamic solution where both the future cash in-flow from distributors and upcoming payment terms (invoices) from the suppliers are unknown. In both the cases, we determine the quality of the heuristic solutions by comparing it with the optimal and lower bound solution of the problem. Lastly we present our managerial implications on applicability of different solution techniques in various real life situations.

2. Related work

The flow of money in a supply chain has not yet attracted the attention of main stream POM and MS/OR researchers even though the problem is important and bears a great resemblance to flow of material. The money flow problem has primarily been studied as the problem of cash circulation, cash management and cash balance. Based on the available literature, the research work under the rubric of financial supply chains can be divided into the following three categories.

- Cash flow systems analogous to ERP systems.
- Models for cash management based on inventory concepts.
- Cross functions models that integrate manufacturing and finance decisions.

2.1. Cash flow systems analogous to ERP systems

There is a pleothra of literature on financial supply chains that has primarily focused on the use of technology in improving the cash flow process similar to that of ERP in a manufacturing environment. The examples include Hausman (2005), Killen (2002) and SAP (2005). The main focus of these studies is on the improvement of actual business process interactions across multiple organizations in financial supply chain systems. For example, Hausman (2005) has studied the process of monetary flow in complex financial relationships involving banks, financial institutes, vendors and retailers. A quantitative measure of the efficiency of cash flow processes can be found in research papers that are termed as “Cash-to-Cash” (C2C) studies. The “Cash-to-Cash” is defined as “the average days required to turn a dollar invested in a raw material into a dollar collected from a customer,” Farris and Hutchison (2002, 2003). Farris and Hutchison (2002) has shown the importance of C2C as a metric in the supply chain system. More recently, Ozbayrak and Akgun (2006) focused on cash conversion cycle which is the time elapsed between the time a purchase or investment is made and the time of sales revenue received from goods produced by that purchase or investment. This approach of cash management may not be applicable in value-added-service operations where it is very difficult to pin point the exact return for each and every purchase and investment. In many cases such purchase

Author's personal copy

and investment are made for strategic advantages, with no immediate clear-cut return. We believe that the flow of cash needs to be managed as an overall problem rather than trying to map which upstream flow results in which downstream flow and then make decisions. Such mapping approach may result in a non-optimal performance of the overall business in terms of cash situations of the company. The studies that deal with cash flow process or the C2C research do not develop a scheme for an optimal or near-optimal management of cash flow in financial supply chain system, as we have done in this paper. They do not optimize the payment schedule. These studies could be considered complimentary to the contribution of this paper because our paper specifies the optimal payment schedule whereas these studies focus on efficient processes.

2.2. Models for cash management based on inventory concepts

Another stream of research in cash management literature has borrowed concepts from inventory theory. In general, an organization maintains a portfolio of assets that include liquid cash, treasury bills, commercial papers, etc. The optimal cash policies for these organizations can be determined by minimizing costs of holding cash and various transaction costs to convert from one asset type to another. The mathematical models for cash balance primarily focus on balancing the cash in hand with the liquid asset like marketable securities based on firms' needs for cash and predictability of such needs. Ashford et al. (1988) and Steuer and Na (2003) are good overviews of applying operational research techniques for financial decision making including both the short term and long term cash management in investments. One of the earliest papers on this topic, Sethi (1971) developed a mathematical programming model for cash management with minimum transaction cost as the objective function. Vangijlswijk (1987) addressed the problem of liquidity position of a firm by carefully balancing transactions across money market, deposits and loans. Mulvey and Vladimirou (1992) presented a networking programming model for asset allocation in portfolio selection and international cash management. Zopounidis (1999) has applied operational research techniques in long term final decisions such as allocation of funds, financial structure and short term decisions such as stocks, account receivables, current liabilities. In Premachandra (2004), a cash management approach has been developed to balance the cash in hand and the assets such as treasury bills, commercial papers, etc. Feinberg and Lewis (2007) proposed solution conditions for optimizing average cost of inventory for the cash balance problem. Gormley and Meade (2007) presented a policy to minimize transaction cost based on cash flow forecast and uncertainty associated with the forecast. Impulse control model has been developed for cash management or money demand in Bar-Ilan et al. (2004). Batlin and Hinko (1982) took a game theoretic approach between two distinct corporate activities – cash flow acceleration and liquidity account allocation. Cash management for randomly varying environment has been studied by Hinderer and Waldmann (2001). The decision problem about when to acquire loans based on cash situations of a company has been studied in Lam et al. (1998). The cash balance problem between the cash fund and short term securities for an infinite time horizon has been addressed by Baccarin (2002).

The cash balance problems addressed in above papers are orthogonal to the problem we are studying. The cash balance problems in these papers deal with internal cash management of an organization so that transaction cost is minimized or higher return can be found from these transactions. However, the problem we are studying focuses on management of external cash transactions such as cash received from downstream partners and cash payables to upstream partners.

2.3. Cross functions models that integrate manufacturing and finance decisions

Some recent research papers (Badell et al., 2004, 2005; Guillen et al., 2007) have emphasized that financial supply chain decisions should be integrated with advanced planning and scheduling decisions. These papers developed mixed integer linear programming based formulations for cash management in a chemical process industry. Cash management problem studied in these papers is based on maximizing the cash position by combining profit and the cost of making that profit. This approach may be applicable for manufacturing industries. However, in service industries such an approach may not be plausible. Our research bears some similarity to the approach presented in these papers. However, we address the problem of cash management to prioritize the payment schedule based on incoming revenue stream and pending invoices to be paid. The results of this study can be applied between any two levels of upstream and downstream partners, in both manufacturing and service industry in a supply chain.

In a recent paper, Rajamani et al. (2006) discuss the Federal Reserve System of USA's cash circulation policy and proposes a framework to analyze the cash supply chain structure. Rajamani et al. (2006) provide a framework for the physical transportation of money, similar to that of flow of materials, in a supply chain. Our model of upstream and downstream flow of cash follows that framework but the flow of money does not have to be in physical form. We focus on incoming and outgoing monetary transactions.

The closest research to the problem we are addressing is Gupta et al. (1987), where authors have developed an integer programming model for the loan payment policies. The problem studied by Gupta et al. (1987) has borrowed concepts from the scheduling literature and is a much simplified version of the problem we are studying in this paper. Vanderknoop and Hooijmans (1989) has also done a similar study by manipulating incoming receipts and outgoing cash-flows. However, many of the realistic characteristics such as deadline, interest, penalty are not considered in these models. Additionally the application of these models require solving complex mathematical programming model. In this paper, we developed a model that is much more concise and practically relevant, and we propose a number of simple heuristics that can be applied in wide variety of situations.

3. Model development

We consider the problem from the viewpoint of a wholesaler who receives finished products from several manufacturers (upstream partners) and then distributes these products to several retailers (downstream partners). The wholesaler receives money from the downstream partners and makes payments to the upstream partners. Let us assume, q_t be the total money received by the wholesaler from all downstream partners at time t.

For tractability purposes, here we are assuming that the future cash in-flows from downstream partners and future invoices along with their terms from upstream partners are known, later in the paper (Sections 4.3 and 4.4) we will relax these constraints to develop a more realistic solution.

We denote L_k to be the invoice amount for the kth invoice from an upstream partner received at time s_k. Note that, at any time a number of invoices may be generated, i.e., there may exist multiple invoices L_k with the same value of s_k. An invoice is generated by a supplier after shipping the products to the wholesaler. The objective of the wholesaler is to schedule the payments of these invoices to the upstream partners within the constraints of the receipt of the money from the retailers. If invoice k is paid before a certain date, denoted as b_k, the terms of payment of the invoice guarantees

Author's personal copy

Table 1
Parameters of the model.

Set
K = set of all invoices

Parameters
L_k = invoice amount for invoice k $\forall k \in K$
u_k = discount rate on the invoice k's amount $\forall k \in K$
b_k = the time on or before which the invoice k needs to paid to get the discount u_k $\forall k \in K$
d_k = due date for the invoice k $\forall k \in K$
υ_k = penalty or interest rate per period if the invoice k is not paid on or before due date d_k $\forall k \in K$
s_k = time at which invoice was generated by the upstream partner $\forall k \in K$, $s_k \leqslant b_k \leqslant d_k$
r = interest rate that can be earned per day for accumulated cash by the wholesaler ($r < \upsilon_k$)
q_t = total amount received from all downstream partners at time t
δ = cash in hand at the beginning time 0

a discount u_k. Existence of such discount can be found in previous researches on cash management such as Borde and McCarty (1998). This discount by upstream partners is given to encourage early payment of the invoice by the wholesaler. However, a penalty or interest υ_k has to be paid if the payment for invoice k is not made within a due date d_k. It may be noted that $s_k \leqslant b_k \leqslant d_k$. Any money that is accumulated with the wholesaler can be invested to earn an interest. The wholesaler's objective is to minimize cash out flow to pay all invoices.

Following, we develop an integer programming model to minimize the net present value of the cash out flow to make payments to the upstream partners. Parameters of the model are given in Table 1.

To formulate the problem in discrete time frame, we consider T as the number of future days representing the planning horizon. We want to decide the day of payment for invoice k $\forall k \in K$ during the planning horizon T. So we have,

Decision variables:

$X_{kt} = 1$, if invoice k is paid on day t, 0 otherwise.

We denote A_k as the amount paid for the invoice k. A_k may take one of the following three values: (i) Invoice k is paid with a discount on or before b_k, i.e., $A_k = L_k(1 - u_k)$. (ii) Invoice k is paid after b_k, but on or before due date d_k, i.e., $A_k = L_k$. (iii) Invoice k is paid at time t after due date d_k with a certain penalty that depends on t. We assume that if the invoice is not paid before the due date, the penalty or the interest starts accruing daily from the due date. So in this case we will have $A_k = L_k(1 + \upsilon_k)^{(t-d_k)}$. Let us denote $A_k(t)$ as the amount paid for invoice k, if the invoice k is paid at time t. So we will have,

$$A_k(t) = \begin{cases} L_k(1 - u_k) & \text{if, } s_k \leqslant t \leqslant b_k \\ L_k & \text{if, } b_k < t \leqslant d_k \\ L_k(1 + \upsilon_k)^{(t-d_k)} & \text{if, } d_k < t. \end{cases}$$

The present value, denoted as PV_k of $A_k(t)$ is $PV_k = A_k/(1 + r)^t$ where r is the interest rate. The wholesaler's objective is to minimize the total present value of the payments made against all invoices, thus

$$\text{Minimize } \sum_{\forall k \in K} PV_k \text{ i.e.,}$$

$$\text{Minimize } Z = \sum_{\forall k \in K}\left[\sum_{t=s_k}^{b_k} \frac{L_k(1-u_k)}{(1+r)^t} X_{kt} + \sum_{t=b_k+1}^{d_k} \frac{L_k}{(1+r)^t} X_{kt} + \sum_{t=d_k+1}^{T} \frac{L_k(1+\upsilon_k)^{(t-d_k)}}{(1+r)^t} X_{kt}\right]. \quad (1)$$

$$X_{kt} = 0 \quad \text{or} \quad 1, \quad \forall k \in K, \quad \forall t \in T. \quad (2)$$

Each invoice needs to be paid only once during the planning horizon T. So we have,

$$\sum_{t=s_k}^{T} X_{kt} = 1, \quad \forall k \in K \quad (3)$$

Also, an invoice cannot be paid before it has been generated, thus,

$$X_{kt} = 0 \quad \forall k \in K \quad \text{and} \quad t < s_k. \quad (4)$$

We also need to specify cash balance equations to ensure that the total cash in hand is more than or equal to the total payments made against one or more invoices in each time interval, i.e., on each day. The cash in hand in each time interval is equal to the total cash inflow received so far, plus the interest earned on the cash-in-hand minus the total payment of invoices made so far. Additionally, we assume that all cash transactions (payment of loans and cash in-flow) occur at the end of the each time interval (e.g., at the end of the day).

We will have following constraints to balance the cash inflow and outflow on each day.

$$\text{For } t = 1, \quad \delta + q_1 - \sum_{\forall k \in K} X_{k1} A_k(1) \geqslant 0.$$

$$\text{For } t = 2, \quad \left[\delta + q_1 - \sum_{\forall k \in K} X_{k1} A_k(1)\right](1 + r) + q_2 - \sum_{\forall k \in K} X_{k2} A_k(2) \geqslant 0.$$

We can generalize the above constraints for all $t = 1, \ldots, T$ as follows.

For a given t,

$$\delta(1 + r)^{t-1} + \sum_{t'=1}^{t}\left[q_{t'}(1 + r)^{t-t'} - \sum_{\forall k \in K} X_{kt'} A_k(t')(1 + r)^{t-t'}\right] \geqslant 0$$
$$\forall t = 1, \ldots, T \quad (5)$$

Thus the scheduling of invoice payments to upstream partners can be represented by an integer programming (IP) formulation P as defined by objective (1) and constraints (2)–(4).

The IP problem presented in this section is NP-Complete as shown below in Theorem 1.

Theorem 1. *Problem P is NP-Complete.*

Proof. The generalized assignment (GA) problem as given below is known to be a NP-Complete problem (Ross and Soland, 1975).

$$\text{Minimize} \quad \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

$$\text{Subject to} \quad \sum_{j \in J} r_{ij} y_{ij} \leqslant e_i, \quad \forall i \in I \quad (6)$$

$$\sum_{i \in I} y_{ij} = 1, \quad \forall j \in J \quad (7)$$

$$y_{ij} = 0 \quad \text{or} \quad 1 \quad (8)$$

subscript$i > 0$, $\forall i \in I$, c_{ij}, $\forall i \in I$ and $j \in J$, r_{ij}, $\forall i \in I$ and $j \in J$ are constants.

The objective function Eq. (1) maps to the objective function of the generalized assignment problem where index t in the problem P maps to index i of GA problem and index k of the problem P maps to index j of the GA problem.

The Eq. (3) of the problem P maps to the Eq. (7) of the GA problem. The Eq. (5) of the problem P maps to the Eq. (6) of the GA problem. Thus the problem P is an instance of the GA problem, which is a NP-Complete problem. So the problem P is also NP-Complete. □

3.1. Additional constraint

One of the tactics in solving NP-Complete IP problem is to introduce additional constraints that will reduce the number of

Author's personal copy

variables in consideration and thus the problem size. Following this, we identify several structural properties of our problem, which we present here as lemmas and correspondingly introduce additional constraints that eliminate some of the decision variables. The detailed discussion of these lemmas and formal proof can be found in Gupta et al. (1987).

Lemma 1. *It is not beneficial to pay an invoice at t, where $t < b_k$.*

Proof. If an invoice k is paid at time $t < b_k$, we will have $A_k(t) = L_k(1 - u_k)$.

If the same invoice k is paid at time $t = b_k$, we will also have $A_k(t) = L_k(1 - u_k)$.

If the invoice k is paid at $t < b_k$, the wholesaler looses the interest on A_k at the rate of r from t to b_k.

So the Lemma 1 follows.

Thus we will have following constraints for the problem P.

$$X_{kt} = 0, \quad \forall k \in K \quad \text{and} \quad t < b_k. \qquad \square \tag{9}$$

Lemma 2. *It is not beneficial to pay an invoice at t, where $b_k < t < d_k$ compare to paying at $t = d_k$.*

Proof. If an invoice k is paid at time t, where $b_k < t < d_k$, we will have $A_k(t) = L_k$.

If an invoice k is paid at time $t = d_k$, we will also have $A_k(t) = L_k$.

If the invoice k is paid at t, where $b_k < t < d_k$, the wholesaler loose the interest on A_k at the rate of r from t to d_k.

So, the Lemma 2 follows.

Thus we will have following constraints for the problem P.

$$X_{kt} = 0, \quad \forall k \in K \quad \text{and} \quad b_k < t < d_k. \qquad \square \tag{10}$$

Lemma 3. *If the total accumulated cash at $t = b_k$ is not sufficient to pay the invoice k, then $X_{kt} = 0$ for $t = b_k$.*

The above Lemma says that if the cash in hand is not sufficient to pay an invoice, the invoice cannot be paid. This is quite obvious and does not require any proof.

However, the lemma helps us to derive a set of additional valid constraints.

The total accumulated cash at the end of any interval t is given by the LHS of Eq. (5),

$$\delta(1+r)^{t-1} + \sum_{t'=1}^{t}\left[q_{t'}(1+r)^{t-t'} - \sum_{\forall k \in K} X_{kt'}A_k(t')(1+r)^{t-t'}\right].$$

If so far no payment is made against any invoice during the time horizon, then we will have $A_k(t') = 0 \quad \forall t' = 1, \ldots, t$.

In this case the total accumulated cash is, $\delta(1+r)^{t-1} + \sum_{t'=1}^{t} q_t(1+r)^{(t-t')}$.

Clearly, this value of the accumulated cash is the upper bound of the accumulated cash at the end of any interval t. So, if an invoice needs to be paid at the interval t, we should have,

$$\delta(1+r)^{t-1} + \sum_{t'=1}^{t} q_t(1+r)^{(t-t')} > A_k(t).$$

The minimum value of $A_k(t)$ is $L_k(1 - u_k)$.

Thus, the invoice k cannot be paid at the interval t if

$$\delta(1+r)^{t-1} + \sum_{t'=1}^{t} q_t(1+r)^{(t-t')} < L_k(1 - u_k).$$

So, we derive a set of valid constraints,

$$X_{kt} = 0 \quad \text{if} \quad \delta(1+r)^{t-1} + \sum_{t'=1}^{t} q_t(1+r)^{(t-t')} < L_k(1 - u_k), \quad \forall k \in K, \quad \forall t \in T. \tag{11}$$

Lemma 4. *If the total accumulated cash at d_k is not sufficient to pay the invoice k, then $X_{kt} = 0$ for $t = d_k$.*

The above lemma is obvious and does not require any proof. However, following the approach in Lemmas 3 and 4 also helps us to derive a set of additional valid constraints as follows:

$$X_{kt} = 0 \quad \text{if} \quad \sum_{t'=1}^{t} q_t(1+r)^{(t-t')} < L_k, \quad \forall k \in K \quad \forall t \in T : t <= d_k. \tag{12}$$

4. Solution approaches

In this section, we present two heuristic solution techniques and the efficacy of their solution results. Ideally the heuristic solution needs to be compared with the IP optimal solution. In this research we use CPLEX 10.0 (ILOG, 2008) for solving the IP optimal. However, the optimal solution of the IP problem P even with the addition of constraints (9)–(12) is impossible to obtain for large size problems. The problem size of the problem P is determined by the number of invoice in considerations ($|K|$) and number of days (T). We have been able to get optimal solution of the IP problem P (along with additional constraints) for up to 30 invoices and 730 days (i.e., 2 years) in a computer with 2GB RAM and Dual core 2.33 GHz Intel processors using CPLEX 10.0. For problem sizes larger than this we need to devise a way to find the lower bound of the problem P, that can be used for judging the quality of solutions obtained by heuristic approach.

4.1. Lower bound determination

Lagrangian Relaxation technique (Fisher, 1981) is a way to relax one of the constraints of an IP problem and introduce a penalty in the objective function for violating the constraint. This modified Lagrangian relaxed problem may be easier to solve than the original IP problem and sometimes helps to get the lower bound of an otherwise difficult IP problem.

For problem P, we relax the constraints denoted by (4) to have the following Lagrangian objective function for some $\lambda_t \geqslant 0 \; \forall t = 1, \ldots, T$.

$$Z(\lambda_1, \ldots, \lambda_T) = \text{Minimize}$$
$$\sum_{\forall k \in K}\left[\sum_{t=s_k}^{b_k} \frac{L_k(1-u_k)}{(1+r)^t}X_{kt} + \sum_{t=b_k+1}^{d_k} \frac{L_k}{(1+r)^t}X_{kt} + \sum_{t=d_k+1}^{T} \frac{L_k(1+v_k)^{(t-d_k)}}{(1+r)^t}X_{kt}\right]$$
$$+ \sum_{t=1}^{T}\lambda_t\left[\delta(1+r)^{t-1} + \sum_{t'=1}^{t}\left[q_t(1+r)^{t-t'} - \sum_{\forall k \in K} X_{kt'}A_k(t')(1+r)^{t-t'}\right]\right]. \tag{13}$$

We denote the Lagrangian relaxed problem P_L with objective function (13) and constraints defined as Eqs. (2)–(4) and (9)–(12). The problem P_L is separable into $|K|$ problems, each of which is independent of $k \in K$. So, each of these problems is easier to solve than the original problem P. There exists non-negative values of $\lambda_t \geqslant 0$ $\forall t = 1, \ldots, T$, such that $Z* \leqslant Z(\lambda_1, \ldots, \lambda_T) \leqslant Z$, where $Z*$ denotes the linear relaxation value of Z (defined by (1)). We apply sub-gradient technique (Fisher, 1985) to find these values of λ using CPLEX 10.0

Author's personal copy

(ILOG, 2008) and Java programming, for which $Z(\lambda_1,\ldots,\lambda_T)$ will be the lower bound of the original problem P.

We use this lower bound value to find the quality of heuristic solutions for large size problems.

4.1.1. Quality of lower bound solution

To demonstrate the quality of lower bound solution we generated several problems for various sizes. The values of different parameters were chosen randomly within a range as denoted in Table 2. Four of these parameters (r, q_t, δ, T) are constant in the model, so there is no maximum value corresponding to these parameters. These range values were chosen based on our discussion with few small businesses in South Florida region in the area of logistic management.

For experimentation, the problem size was varied by changing the number of invoices to 10, 15, 20, 25 and 30. For each invoice number we generated 5 problems by having different random seed. So we had a total of 25 problems. For each of these problems we found the lower bound solution in CPLEX 10.0 in the machine configuration mentioned before. We also obtained the optimal solution using CPLEX 10.0. As mentioned before, we have been able to get the optimal solution up to problem size of 30 invoices. We compute the maximum, minimum and average gap of the lower bound from the optimal solution across five problems for each problem size. The percentage gap for the lower bound solution with respect to optimal solution is computed as,

$$(\text{Optimal solution} - \text{lower bound solution}) \times 100/\text{optimal solution}$$

and reported in Table 3. The results demonstrate that the lower bound obtained by Lagrangian Relaxation technique is very near to the optimal within on 0.1%. For larger problem sizes, we cannot find the optimal solution, so there is no way to judge the quality of Lagrangian Relaxation solution for large size problems. However, near optimal result for Lagrangian Relaxation gives us certain degree of confidence in measuring the solution quality of our heuristic techniques (to be discussed in next few subsections) by comparing the solution with lower bound found by Lagrangian Relaxation.

Table 2
Range values for model parameters.

Parameter	Minimum value	Maximum value
L_k	\$500.00	\$10000.00
b_k	10 Days	30 Days
u_k	1%	5%
d_k	10 Days	90 Days
v_k	0.02% Daily interest i.e., 7.0% APY	0.1% Daily interest i.e., 44% APY
s_k	0	180 Days (6 months)
r	0.01% Daily interest i.e., 3.7% APY	Constant
q_t	\$500.00	Constant
δ	0	Constant
T	730 Days i.e., 2 Years	Constant

Table 3
Lower bound gap compared to IP optimal solution.

Number of invoices	% Min gap	% Max gap	% Average gap
10	0.02	0.09	0.05
15	0.01	0.08	0.04
20	0.02	0.08	0.05
25	0.03	0.12	0.06
30	0.01	0.11	0.05

4.2. Interval heuristic

One dimension of the complexity of the problem P is the number of time intervals T, i.e., number of days in consideration. In Interval Heuristic, we merge several time intervals into a single time interval, e.g. a number of days together may compose a week or months and reformulate the problem as a new problem P^H with reduced number of intervals. This will reduce the problem complexity significantly and will enable us to solve P^H optimally. The optimal solution of P^H will be a heuristic solution for the problem P.

Thus given a problem P, we create a new problem P^H, that is obtained by merging n number of time intervals of problem P into one interval. Thus, if $n = 7$, in problemP^H, one time interval is a week.

The number of time periods in P^H will be $T^H = \lceil T/n \rceil$. The ceiling takes care of the scenario where T is not exactly divisible by n. In this case, the length of the last interval (i.e., Tth interval), will be the remainder of T/n.

The interest rate for each interval in P^H will be computed as compound interest rates for n intervals in problem P. The interest for the last interval in problem P^H also needs to be adjusted based on the length of the last interval, which is $\rho = T\%n$, where % denotes the remainder of T/n. Thus in problem P^H, interest rate on accumulated cash at wholesaler

$$r_t^H = \begin{cases} (1+r)^n - 1 & \forall t = 1,\ldots,(T^H - 1), \\ (1+r)^\rho - 1, & t = T^H, \quad \text{where} \quad \rho = T\%n \end{cases}.$$

The amount received from downstream partner in an interval in problem P^H is also adjusted accordingly as follows.

$$q_t^H = \begin{cases} \sum_{\hat{t}=tn}^{tn+n-1} q_{\hat{t}} & \forall t = 1\ldots(T^H - 1) \\ \sum_{\hat{t}=tn}^{tn+\rho-1} q_{\hat{t}}, & t = T^H, \quad \text{where} \quad \rho = T\%n \end{cases}.$$

Similarly, other parameters of problem P^H are computed as, $L_k^H = L_k\ \forall k \in K$, $u_k^H = u_k\ \forall k \in K$, $s_k^H = \lfloor s_k/n \rfloor\ \forall k \in K$, $b_k^H = \lfloor b_k/n \rfloor\ \forall k \in K$, $d_k^H = \lfloor d_k/n \rfloor\ \forall k \in K$, where $\lfloor\ \rfloor$ denotes the floor function of a number. Following the same logic as in case of r, we compute the penalty rate per interval for problem P^H by compounding over multiple intervals as, $v_{kt}^H = \begin{cases} (1+v_k)^n - 1 & \forall t = 1,\ldots,(T^H - 1) \\ (1+v_k)^\rho - 1 & \text{for } t = T^H, \text{ where } \rho = T\%n \end{cases}.$

The problem P^H is formulated as an instance of problem P, by Eq. 1, where v_k will be replaced by v_{kt}^H, q will be replaced by q_t^H and r will be replaced by r_t^H. All other parameters will have a suffix H as defined above.

The solution of problem P^H will give us a heuristic solution for the problem P. For $n = 1$, problem P^H reduces to the original problem P. As n is increased the accuracy of the heuristic solution will reduce, however with small n it will be difficult to solve P^H optimally. So, there need to be a tradeoff between the accuracy and the problem complexity. In the next section we compare the variation of accuracy and the complexity of the problem P^H as n is varied.

4.2.1. Quality of interval heuristic

In this section we first compare the solution obtained by interval heuristic with that of the optimal solution. For problems of sizes 10, 20, 30, 40 and 50 number of invoices, we created 5 problems each with parameter ranges as described in Table 2. For each of these problems we found the solution by interval heuristic with interval length $N = 7$ (i.e., one week). For problem size of up to 30

Author's personal copy

Table 4
% Gap for interval heuristic solution.

Problem size (number of invoices)	% Gap with optimal solution			% Gap with lower bound solution		
	Min	Max	Average	Min	Max	Average
10	0.07	0.14	0.13	0.13	0.21	0.12
20	0.20	0.17	0.15	0.16	0.22	0.22
30	0.11	0.22	0.19	0.18	0.27	0.22
40	–	–	–	0.12	0.23	0.19
50	–	–	–	0.19	0.30	0.21

invoices, we found both the optimal solution and the lower bound solution as described in Section 4.1. However, for problem sizes of 40 and 50 invoices, we could not compute the optimal solution, but computed the lower bound solution as described in Section 4.1. We computed the quality of the heuristic solution by calculating the % Gap of the heuristic solution from the optimal solution and the lower bound (LB) solution as follows.

$$\%\ \text{Gapwithoptimalsolution} = (\text{heuristicsolution} - \text{optimalsolution}) \times 100/\text{optimalsolution}, \quad (14)$$

$$\%\ \text{GapwithLBsolution} = (\text{heuristicsolution} - \text{LBsolution}) \times 100/\text{LBsolution}. \quad (15)$$

We report this gap in Table 4. The results demonstrate that interval heuristic, with 7 days (i.e., one week) defined as one interval, gives result within 0.3% of the optimal solution.

Intuitively, with the increase in number of intervals (n) the gap between the interval heuristic and the optimal solution will increase, whereas it will be easier to solve interval heuristic with higher n. To demonstrate the tradeoff between the accuracy and the solution complexity, we next study how the accuracy and solution complexity change with the number of intervals (n). The quality of the Interval Heuristic solution will be measured by % Gap as defined in (14). The complexity or hardness to solve the interval heuristic problem optimally will be measured by the time (in seconds) CPLEX 10.0 takes to solve the problem. We select problem size of 30 invoices, the maximum size we can solve optimally, for this purpose. For this problem size we generate 5 problem instances by different random seeds. The values of all other parameters are chosen as defined in Table 2. For each of these problems, we compute the optimal solution using CPLEX 10.0 in the machine configuration described in Section 4. For each problem, we also create interval heuristic problems for interval size n = 3, 5, 7, 14, 21, 28 and 30. So, we had a total of 35 interval heuristic problems, 5 problems for each interval size. We solve these interval heuristic problems optimally using CPLEX 10.0. We note down the time CPLEX takes to solve each of these problems. For each interval size, we first compute the average % Gap from optimal solution and the average time CPLEX takes to generate solution for the interval heuristic problem. To plot in the same graph and compare, we normalize the % Gap and solution time within [0, 1] range. We report this normalized value of Gap (indication of quality) and the solution time (indication of complexity of the interval heuristic problem) on Y-axis along with the number of intervals (n) on X-axis in Fig. 2.

With an increase in the value of n, the complexity of the problem P^H reduces and so does the solution time. From Fig. 2, we can see that as n increases from 2 to 7, the time to solve the problem decreases, implying that the complexity of the problem reduces. However beyond n = 7 there is no appreciable decrease in the solution time. On the other hand, with the increase of n, the approximation of considering a set of original time periods as an interval

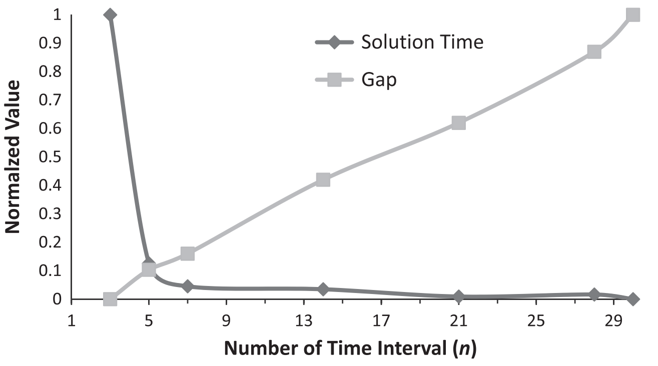

Fig. 2. Tradeoff between complexity (solution time) and accuracy (Gap) with number of intervals in interval heuristic.

reduces the accuracy of the solution. Following this, Fig. 2 demonstrates that the % Gap increases almost linearly with n, throughout all the values of n.

An interval of 7 days or one week in the interval heuristic will help in solving larger problems with fewer resources with a marginal loss (0.16%) in the quality of the solution. However, there is not much gain in increasing n beyond one week.

Thus as demonstrated, the interval heuristic will perform poorly if each interval is composed of more than 7 days. Additionally, the interval heuristic requires solving the reduced size problem optimally. However, for a very large size problem, where the number of invoices is in the range of hundreds, even the interval heuristic problem becomes impossible to solve optimally. In such instances we need to resort to heuristics as discussed in next two sections.

4.3. Dynamic invoice in-flow problem

In the above sections we considered a static problem in which the number of invoices to be paid, the amounts of these invoices, and the amount of money received from the downstream partners are known when the payment schedule is being prepared. The implicit assumption is that no other invoices will be received (or considered for payment) and there will be no cash inflow from downstream partners until the given set of the invoices has been paid. However, the real life situations are dynamic where invoices constantly flow in from upstream partners to the wholesaler and cash is being received from the downstream partners. In such situations we need to decide which of the pending invoices have to be paid on a given day. The set of pending invoices is dynamic in nature because of the receipt of new invoices. In this section we develop a heuristic that can accommodate the scenario where invoice flow is dynamic, but the future cash in-flow (q_t) is known before hand.

To develop this heuristic, we use adjacent pair-wise interchange to determine the sequence in which two invoices j and k out of the set of unpaid invoices will be paid on a given day (see Fig. 3). We start by considering the set of K invoices in the static environment and develop the dominance rules. Partition the set K such that $K = \{B\ j\ k\ A\}$ where B is the subset of invoices that have already been sequenced to be paid and A is subset of invoices that will be paid after invoice j and k have been paid. We derive dominance conditions under which invoice j will be paid before k.

Let the time at which the last invoice in B was paid is $h - 1$. The current time is, therefore, $t = h$, where the wholesaler has two invoices to pay, j and k, such that $j,k \in K - B$. The wholesaler has two choices: (I) pay j first and then k, and (II) pay k first and then j. Let us also assume that β, the accumulated cash at $t = h$, and the additional cash accumulation after $t = h$ will be used to pay either invoice j or k. Let t_j and t_k be the times at which

Author's personal copy

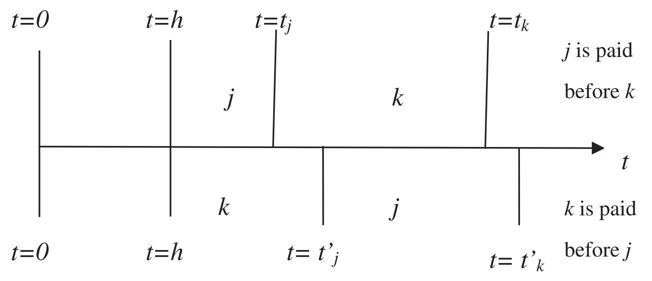

Fig. 3. Comparing two loans j and k.

invoices j and k are paid respectively in case I. Similarly, let t'_j and t'_k be the times at which invoices j and k are paid respectively in case II. For this analysis to have the tractability we assume $q_t = q$ $\forall t$. In case I, we will have, $A_j = \beta(1+r)^{(t_j-h)} + q\left((1+r)^{(t_j-h)} - 1\right)/r$ and $A_k = q\left((1+r)^{(t_k-t_j)} - 1\right)/r$. Thus the present value of total payments made is,

$$PV_{jk} = \frac{A_j}{(1+r)^{t_j}} + \frac{A_k}{(1+r)^{t_k}} = \frac{\beta(1+r)^{(t_j-h)} + q\left((1+r)^{(t_j-h)} - 1\right)}{r(1+r)^{t_j}} + \frac{q\left((1+r)^{(t_k-t_j)} - 1\right)}{r(1+r)^{t_k}}. \quad (16)$$

Similarly, for case II, the present value of total payment made is,

$$PV_{kj} = \frac{A'_j}{(1+r)^{t'_j}} + \frac{A'_k}{(1+r)^{t'_k}} = \frac{\beta(1+r)^{(t'_j-h)} + q\left((1+r)^{(t'_j-t'_k)} - 1\right)}{r(1+r)^{t'_j}} + \frac{q\left((1+r)^{(t'_k-h)} - 1\right)}{r(1+r)^{t'_k}}. \quad (17)$$

If we assume, it is beneficial to pay j first and then k, we will have $PV_{jk} < PV_{kj}$, thus,

$$\frac{\beta(1+r)^{(t_j-h)} + q\left((1+r)^{(t_j-h)} - 1\right)}{r(1+r)^{t_j}} + \frac{q\left((1+r)^{(t_k-t_j)} - 1\right)}{r(1+r)^{t_k}} < \frac{\beta(1+r)^{(t'_j-h)} + q\left((1+r)^{(t'_j-t'_k)} - 1\right)}{r(1+r)^{t'_j}} + \frac{q\left((1+r)^{(t'_k-h)} - 1\right)}{r(1+r)^{t'_k}} \quad (18)$$

From this we can derive that, wholesaler will pay invoice j first and then k if and only if $t_k < t'_j$. Thus, we will have following Lemma.

Lemma 5. *Present value of total payment against invoices will be minimal if the total duration to pay a set of invoices can be minimized.*

We compute t_j, by solving equations,

$$\begin{aligned} q\left((1+r)^{(t_j-start_j)} - 1\right)/r &= L_j(1-u_j), && \text{if } t_j \leqslant b_j, \\ q\left((1+r)^{(t_j-start_j)} - 1\right)/r &= L_j, && \text{if } b_j < t_j \leqslant d_j, \\ q\left((1+r)^{(t_j-start_j)} - 1\right)/r &= L_j(1+v_j)^{(t_j-d_j)}, && \text{if } d_j < t_j, \end{aligned}$$

where $start_j = \begin{matrix} h & \text{if } s_j \leqslant h \\ s_j & \text{otherwise} \end{matrix}.$

We use the value of t_j to compute t_k, by solving equations,

$$\begin{aligned} q\left((1+r)^{(t_k-start_k)} - 1\right)/r &= L_k(1-u_k), && \text{if } t_k \leqslant b_k, \\ q\left((1+r)^{(t_k-start_k)} - 1\right)/r &= L_k, && \text{if } b_k < t_k \leqslant d_k, \\ q\left((1+r)^{(t_k-start_k)} - 1\right)/r &= L_k(1+v_k)^{(t_k-d_k)}, && \text{if } d_k < t_k, \end{aligned}$$

where $start_k = \begin{matrix} h & \text{if } s_k \leqslant h \\ s_k & \text{otherwise} \end{matrix}.$

Similarly, we can compute the values of t'_k and then t'_j. We use the above equations and the insight derived from Lemma 5 to develop the heuristic algorithm in Table 5.

For each pair of invoices the algorithm compares based on Lemma 5 to decide which invoice is better to pay first. Based on all such comparisons, the algorithm identifies the invoice to pay next. The algorithm outputs this invoice as j. The algorithm tries to minimize the total payment duration.

The algorithm starts with the ordering of invoices in list M by the earliest time at which each of the invoices can be paid, if cash accumulation for that invoice starts immediately. In line 6, the algorithm picks up the median invoice (j) in the list M. It then compares all other invoices with the invoice j (Line 8–12), to find out which of the invoices is better to pay before j, and creates a list S of such invoices. Next, in Lines 17 and 18, assigning S to M, the algorithm continues until and unless the algorithm has identified invoice j, such that there is no other invoice that is beneficial to pay before j. This approach of finding invoice j, is very similar to binary sort approach that enables us to identify j in $O(\log|M|)$ time.

If the invoice j cannot be paid by the accumulated cash at the present time t_1, the cash accumulation for j starts at t_1 and j will be paid in future at time t_2. In between t_1 and t_2, if a new invoice l comes at a time t_3 ($t_1 < t_3 < t_2$), the invoice j and l are compared to find out which invoice is better to pay first. Because of transitivity of the Eq. (18), we do not need to compare the newly arrived invoice l, with all other invoices.

At the time, when one of the invoice is paid, i.e., if j is paid first then at t_2, the algorithm is run again to decide the next invoice to pay.

The algorithm would work well in scenarios where all cash-in flows are known before hand. Note that though for the sake of tractability, in the above algorithm we have assumed that the q_t is same for all future time periods, this is not a requirement. As long as, q_t is known for all future time periods, the equation corresponding to Lemma 5 can be solved and thus the algorithm in Table 5 can be run. The algorithm will also perform well for limited number of future time periods. If all invoices are known before hand for distant future, this algorithm will not perform well.

Complexity of dynamic invoice in-flow heuristic algorithm: The dynamic Invoice In-Flow heuristic algorithm will repeat from once for each invoice, i.e., $|K|$ times. In each execution, the algorithm will find the invoice j in $O(\log|M|)$ time. Thus the overall complexity of the dynamic invoice in-flow heuristic algorithm is $O(|K|Log|M|)$, i.e., log-linear, which is very time efficient. This allows the algorithm to be adapted for dynamic situations as described below.

Table 5
Heuristic for dynamic invoice in-flow.

```
1.  Input: A set of invoice K with parameters as defined in Table 1.
2.  Output: Invoice j that will be paid at the end of time interval t.
3.  Ordered List M = {k: k ∈ K and k has not been paid}, order by the earliest
    time by which invoice k can be paid.
4.  If M = Φ
5.      End.
6.  Let $j = \lceil \frac{|M|}{2} \rceil^{th}$ invoice in the list M
7.  Set S = Φ
8.  For $\forall k \in L : k \neq j$
9.      Compute $t_j$, $t_k$, $t'_j$ and $t'_k$
10.     If $t_k > t'_j$ /* Invoice k to be paid before j */
11.         S = S ∪ {k}
12.     End If
13. End For
14. If S = Φ
15.     Return j
16. Else
17.     M = S, order by the earliest time by which invoice k ∈ S can be paid
18.     Go to line 6
19. End If
```

Author's personal copy

4.4. Dynamic invoice and dynamic cash in-flow problem

In previous section, we developed the heuristic that can be applied in scenarios where invoice flow is dynamic, but the future cash in-flow (q_t) is known before hand. To address situations, where both the future cash in-flow and the invoice flow is dynamic and unknown, we develop an alternate heuristic as shown in Table 6.

The heuristic in Table 6 is run at the end of each time interval t, to determine a set of invoices that can be paid at the end of interval t. The algorithm is based on the assumption that invoices that have crossed the deadline need to be paid first, then the invoices that have crossed the discount date (b_k) but not the deadline (d_k). Also, among the invoices whose deadline have crossed, the algorithm will arrange for payment of the invoices in the descending order of the penalty (v_k).

The above algorithm is based on the assumption that it is always better to pay the invoices that have crossed the deadline. However, if the penalty for crossing the deadline of an invoice is very small (i.e., v_j is small), it may be beneficial to hold the payment of that invoice to pay for other invoices before the deadline, where penalty is high. In such cases, the heuristic decision of the algorithm will not perform well. Thus, if the variance in penalty of invoices is high, the result of the algorithm will not be as good as in other scenarios.

Table 6
Heuristic for dynamic invoice and dynamic cash in-flow.

```
 1. **Input** : A set of invoice K with parameters as defined in Table 1.t, the
    present time interval.
 2. **Output**: £, the set of invoices that will be paid at the end of the present
    time interval t.
 3. Let $F_1$ = Set of invoices for which $t = b_j, j \in F_1$
 4. Let $F_2$ = Set of invoices for which $t = d_j, j \in F_2$
 5. Let $F_3$ = Set of invoices for which $t > d_j, j \in F_3$
 6. Initially Set $F_1 = \Phi$, $F_2 = \Phi$, $F_3 = \Phi$ and £ = $\Phi$
 7. Let $Y \subseteq K$, be the set of all unpaid invoices at the beginning of t.
 8. Let $A_t$= Accumulated cash at the end of t.
 9. For each $j \in Y$,
10.   If $t < b_j, j \notin F_1, j \notin F_2, j \notin F_3$
11.   Else if $b_j < t < d_j, j \notin F_1, j \notin F_2, j \notin F_3$
12.   Else if $t = b_j$ and $L_j(1 - u_j) \leqslant A_t$, $F_1 = F_1 \cup \{j\}$
13.   Else if $t = d_j$ and $L_j \leqslant A_t$, $F_2 = F_2 \cup \{j\}$
14.   Else if $L_j(1 + v_j)^{t-d_j} \leqslant A_t,\ F_3 = F_3 \cup \{j\}$
15. End For
16. If $F_3 \neq \Phi$ AND $A_t > 0$
17.     Order $F_3$ in the descending order of $v_j, j \in F_3$
18.     Select first l invoices in ordered $F_3$, such that $\sum L_j(1 + v_j)^{t-d_j} \leqslant A_t$
19.     $A_t = A_t - \sum L_j(1 + v_j)^{t-d_j}$ and include these l invoices in £
20. End if
21. If $F_2 \neq \Phi$ AND $A_t > 0$
22.     Order $F_2$ in the descending order of $v_j, j \in F_2$
23.     Select first l invoices in ordered $F_2$, such that $\sum L_j \leqslant A_t$
24.     $A_t = A_t - \sum L_j$ and include these l invoices in £
25. End if.
26. If $F_2 \neq \Phi$ AND $A_t > 0$
27.     Order $F_1$ in the descending order of $u_j, j \in F_1$
28.     Select first l invoices in ordered $F_1$, such that $\sum L_j(1 - u_j) \leqslant A_t$
29.     $A_t = A_t - \sum L_j(1 - u_j)$ and include these l invoices in £
30. End if.
```

4.5. Quality of dynamic heuristics

We compare the present value of the total payment obtained by the two dynamic heuristic approaches with that of the optimal solution and the lower bound obtained by Lagrangian Relaxation solution. We considered 10 problem sizes with 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100 invoices for 5 years (i.e., 1825 days). For each problem sizes we generated 5 problem instances with all other parameters randomly selected within the range defined in Table 1. We report the quality of both the dynamic heuristic solution with that of the optimal and lower bound solution as obtained by Eqs. (14) and (15) in Table 7. Due to larger number of days (1825 days), optimal solution can only be found for problems with only 10 and 20 invoices.

Table 7 shows that dynamic invoice in-flow heuristic approach can give solutions within 1.8% of the lower bound solution, whereas dynamic invoice and dynamic invoice and dynamic cash in-flow heuristic can give solutions within 3% of the lower bound solution.

To demonstrate the time efficiency of both dynamic heuristics, we compare the time taken by both dynamic heuristics with the interval heuristic as the problem size is varied by varying the number of invoices and the number of years. We consider 5 problem sizes as shown in Table 8. For each problem sizes, we generated 5 problem instances by varying the random seed. All parameters are chosen randomly within the ranges as defined in Table 1. We run the interval heuristic with 7 days interval and note the time

Table 8
Time efficiency of dynamic heuristic.

Problem size		Average solution time (ms)		
Number of invoices	Number of years	Interval heuristic with seven days interval	Dynamic invoice in-flow heuristic	Dynamic invoice and dynamic cash in-flow heuristic
20	1	56,781	20	8
40	2	130,190	25	9
60	3	320,891	26	11
80	4	601,801	31	14
100	5	141,879	33	15

Table 7
% Gap for dynamic heuristic solution.

Problem size (number of invoices)	Dynamic invoice in-flow heuristic						Dynamic invoice and dynamic cash in-flow heuristic					
	% Gap with optimal solution			% Gap with lower bound solution			% Gap with optimal solution			% Gap with lower bound solution		
	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg	Min	Max	Avg
10	0.78	1.22	0.79	0.72	1.22	0.90	1.19	1.45	1.20	1.33	1.67	1.39
20	0.79	1.29	0.91	0.81	1.32	0.95	1.29	1.59	1.36	1.31	1.69	1.42
30	–	–	–	0.85	1.32	1.19	–	–	–	1.39	1.48	1.61
40	–	–	–	1.08	1.31	1.22	–	–	–	1.59	2.24	1.95
50	–	–	–	1.23	1.63	1.45	–	–	–	1.89	2.46	2.11
60	–	–	–	1.21	1.78	1.40	–	–	–	1.90	2.29	2.21
70	–	–	–	1.29	1.67	1.45	–	–	–	2.01	2.56	2.26
80	–	–	–	1.20	1.61	1.51	–	–	–	2.13	2.78	2.30
90	–	–	–	1.49	1.73	1.59	–	–	–	2.23	2.89	2.43
100	–	–	–	1.42	1.67	1.78	–	–	–	2.39	2.98	2.61

Author's personal copy

taken to find the solution. We also run both the dynamic heuristics and note the time. We report the average time taken by interval heuristic and both dynamic heuristics for each problem size. As demonstrated in Table 8, both dynamic heuristics are more time efficient than the interval heuristic.

5. Discussion and conclusion

We have computationally demonstrated the accuracy and applicability of various heuristics for the cash management problem discussed in this paper. For a 30 invoice problem the interval heuristic for the static problem gave results that were 0.19% higher than the optimal and 0.22% higher than the lower bound. For a 50 invoice problem the percentage gap from the lower bound solution was 0.21%.

The study was then extended to the dynamic problems in which the invoices and payments continue to arrive and are not fixed at the beginning of the planning period. At any given point in time the decision maker has to decide which invoice should be paid at that time. The choice is influenced by the pending invoices and the available cash. We have considered two different scenarios of the dynamic problem. The first one deals with the situation in which the arrival of invoices is dynamic, but the future in-flow of cash is known. In the second case both the arrival of invoices and receipt of cash are dynamic. Two different heuristics are proposed for these two cases. The heuristic results were compared with the lower bound. The percentage gap from the lower bound was 1.78% for the dynamic invoice in-flow problem and 2.61% for the problem in which both the arrival of invoices and receipt of cash are dynamic. The dynamic heuristics are much more efficient computationally as compared to the interval heuristic. Therefore, for very large problems of several years and hundreds of invoices the dynamic heuristic may be preferred over the interval heuristic.

The problem studied in this paper has implications for both managers and researchers. The heuristics proposed in this paper can be used to solve real life dynamic problems. The research work in this paper is rather new and opens up new research area in the study of financial supply chains. The research may be expanded to multiple echelons involving several currencies.

References

Agrawal, S., 2008. Private Communication. CEO, American Medical Depot, Miami, Florida, USA.

Ashford, R.W., Berry, R.H., Dyson, R.G., 1988. Operational-research and financial management. European Journal of Operational Research 36 (2), 143–152.

Baccarin, S., 2002. Optimal impulse control for cash management with quadratic holding-penalty costs. Decisions in Economics and Finance 25, 19–32.

Badell, M., Romero, J., Huertas, R., Puigjaner, L., 2004. Planning, scheduling and budgeting value-added chains.

Badell, M., Romero, J., Puigjaner, L., 2005. Optimal budget and cash flows during retrofitting periods in batch chemical process industries. International Journal of Production Economics 95 (3), 359–372.

Bar-Ilan, A., Perry, D., Stadje, W., 2004. A generalized impulse control model of cash management. Journal of Economic Dynamics and Control 28 (6), 1013–1033.

Batlin, C.A., Hinko, S., 1982. A game theoretic approach to cash management. Journal of Business 55 (3), 367–381.

Borde, S.F., McCarty, D.E., 1998. Determining the cash discount in the firm's credit policy: an evaluation. Journal of Financial and Strategic Decisions 11 (2), 41–49.

Farris II, T.M., Hutchison, D.P., 2002. Cash-to-cash: the new supply chain management metric. International Journal of Physical Distribution and Logistics Management 32 (4), 288–298.

Farris II, T.M., Hutchison, D.P., 2003. Measuring cash-to-cash performance. International Journal of Logistics Management 14 (2), 83–92.

Feinberg, E.A., Lewis, M.E., 2007. Optimality inequalities for average cost Markov decision processes and the stochastic cash balance problem. Mathematics of Operations Research 32 (4), 769–783.

Fisher, M.L., 1981. The Lagrangian relaxation method for solving integer programming problems. Management Science 27 (1), 1.

Fisher, M.L., 1985. An applications oriented guide to Lagrangian relaxation. Interfaces 15 (2), 10–21.

Gormley, F.M., Meade, N., 2007. The utility of cash flow forecasts in the management of corporate cash balances. European Journal of Operational Research 182 (2), 923–935.

Guillen, G., Badell, M., & Puigjaner, L. 2007. A holistic framework for short-term supply chain management integrating production and corporate financial planning.

Gupta, S.K., Kunnathur, A.S., Dandapani, K., 1987. Optimal repayment policies for multiple loans. OMEGA International Journal of Management Science 15 (4), 323–330.

Hausman, H.W., 2005. Financial Flows and Supply Chain Efficiency. Visa Commerical Solutions.

Hinderer, K., Waldmann, K.H., 2001. Cash management in a randomly varying environment. European Journal of Operational Research 130 (3), 468–485.

ILOG, 2008. IBM ILOG Optimization. <http://www-01.ibm.com/software/integration/optimization/cplex/> (Retrieved 2009).

Killen, A., 2002. Optimizing the Financial Supply Chain. Killen and Associates, Palo Alto.

Kouvelis, P., Chambers, C., Wang, H., 2006. Supply chain management research and production and operations management: review, trends, and opportunities. Production and Operations Management 15 (3), 449.

Lam, K.C., Runeson, G., Tam, C.M., Lo, S.M., 1998. Modelling loan acquisition decisions. Engineering, Construction and Architectural Management 5 (4), 359–375.

Mulvey, J.M., Vladimirou, H., 1992. Stochastic network programming for financial-planning problems. Management Science 38 (11), 1642–1664.

Ozbayrak, M., Akgun, M., 2006. The effects of manufacturing control strategies on the cash conversion cycle in manufacturing systems. International Journal of Production Economics 103 (2), 535–550.

Premachandra, I.M., 2004. A diffusion approximation model for managing cash in firms: an alternative approach to the Miller–Orr model. European Journal of Operational Research 157 (1), 218–226.

Rajamani, D., Geismar, H.N., Sriskandarajah, C., 2006. A framework to analyze cash supply chains. Production and Operations Management 15 (4), 544–552.

Ross, G.T., Soland, R.M., 1975. A branch and bound algorithm for the generalized assignment problem. Mathematical Programming 8 (1), 91–103.

Sethi, S.P., 1971. A note on a planning horizon model of cash management. The Journal of Financial and Quantitative Analysis 6 (1), 659–664.

SAP, 2005. Financial Supply Chain Management with SAP. SAP Inc.

Steuer, R.E., Na, P., 2003. Multiple criteria decision making combined with finance: a categorized bibliographic study. European Journal of Operational Research 150 (3), 496–515.

Vanderknoop, H.S., Hooijmans, F.C., 1989. Optimal allocation of payments and receipts. European Journal of Operational Research 43 (2), 161–167.

Vangijlswijk, V., 1987. Evaluating the interest aspect of cash management. European Journal of Operational Research 30 (1), 85–88.

Zopounidis, C., 1999. Multicriteria decision aid in financial management. European Journal of Operational Research 119 (2), 404–415.