03/2002/100057

Dynamic Lot Size Problems with One-way Product Substitution[1]

Vernon Ning Hsu

School of Management
George Mason University
Fairfax, VA 22030, USA
email: *vhsu@gmu.edu*

Chung-Lun Li[2]

Department of Logistics
The Hong Kong Polytechnic University
Kowloon, Hong Kong
email: *lgtclli@polyu.edu.hk*

Wen-Qiang Xiao

Graduate School of Business
Columbia University
New York, NY 10027, USA
email: *wx2004@columbia.edu*

December 2002
Revised: November 2003
Revised: June 2004

[1] This research was supported in part by the Research Grants Council of Hong Kong under grant number PolyU6132/02E

[2] Corresponding author

03/2002/100057

Dynamic Lot Size Problems with One-way Product Substitution

Abstract

We consider two multi-product dynamic lot size models with one-way substitution, where the products can be indexed such that a lower-index product may be used to substitute the demand of a higher-index product. In the first model, the product used to meet the demand of another product must be physically transformed into the latter and incur a conversion cost. In the second model, a product can be directly used to satisfy the demand of another product without requiring any physical conversion. Both problems are computationally intractable in general. We develop dynamic programming algorithms that solve the problems in polynomial time when the number of products is fixed. A heuristic is also developed, and computational experiments are conducted to test the effectiveness of the heuristic and the efficiency of the optimal algorithm.

1. Introduction

This paper considers a finite horizon, multi-product, dynamic lot size (DLS) problem with one-way product substitution. There are m products, each with known demands in an n-period planning horizon. The products are indexed such that a lower-index product may be used to substitute for a higher-index product. We consider two types of product substitution: substitution with conversion (SWC) and substitution without conversion (SWO).

In an SWC problem, a lower-index product must be converted through physical transformation to substitute the demand of another higher-index product. In an SWO problem, a lower-index product can be directly used to satisfy the demand of a higher-index product without requiring any physical transformation. The difference between SWC and SWO, from the modeling standpoint, is that in an SWC problem, the substitution takes place when a lower-index product is converted into a higher-index product. If the converted product is not immediately used to satisfy demand, it becomes the inventory of the (converted) higher-index product. In an SWO problem, however, substitution always takes place when a lower-index product is used to satisfy the demand of a higher-index product.

One-way product substitution problems both with and without conversion have been the focus of many studies, most of which have been motivated by real world applications. The benefits of product substitution have long been recognized in these studies. For example, allowing the production and inventory of a product to be used to meet the demand of another (lower grade) product may offer opportunities for economies of scale in managing the cycle stocks of the former. Another significant benefit of product substitution is the possibility of inventory pooling to hedge against demand uncertainties and to help reduce safety stocks.

Similar to the study of many other production planning and inventory problems, the research on one-way inventory substitution problems in the literature follows two streams: Stochastic models deal with demand and supply uncertainties and study issues such as the effect of inventory pooling with product substitution, while deterministic models investigate the dynamics of production and cycle inventory with the presence of economies of scale. Examples of single-period stochastic inventory models for one-way product substitution include recent works by Bassok *et al.* (1999), Hsu and Bassok (1999), Smith and Agrawal (2000), and Rao *et al.* (2004). Smith and Agrawal (2000) have also provided a thorough review on the study of product substitution in the literature in various contexts such as retailing, yield management in the airline industry, and resource allocation.

We now offer a more detailed discussion on the other stream of research on deterministic models, which is closely related to the problem studied in this paper. Drezner *et al.* (1995) and Gurnani and Drezner (2000) discussed an application of the SWC problem, where a more generic product can be transformed (through customization or upgrading) into another less generic product. For example, a retailer could break a larger package of a product to substitute for a shortage of the same product packaged in smaller sizes. They assumed that value is added in the transformation, and therefore, the inventory holding cost of the more generic product is less than that of the less generic product. They formulated the problem as a continuous-time economic order quantity (EOQ) model and obtained optimal order quantities for all the products. Swaminathan and Kucukyavuz (2001) applied the SWC problem to the biotechnology industry. In their application, a reagent product used for DNA amplification can be carried in three forms. The reagent product in a more bulky form can be converted and packaged into another more specialized packaging form. However, unlike the above retailer's example in which the conversion involves simple repackaging, converting a form of reagent packaging into another may require a complicated and costly process in which the reagent has to be synthesized. The problem is formulated both as a continuous-time EOQ model and a discrete-time DLS model. The DLS model, which deals with additional constraints such as product life and maximum number of conversions, assumes stationary setup, holding, and conversion costs, and is solved as a constrained shortest path problem.

Examples of SWO can be found in many applications such as retailing (substituting a product for which there is a shortage with another similar product), airline ticket booking (substituting a lower-class seat with a higher-class seat), and manufacturing (substituting a lower-quality component with a higher-quality component). Pentico (1974, 1976) studied an SWO problem in the retailing industry, where a (higher-grade) product can be used to substitute for another (possibly lower-grade) product that is in short supply. Pentico formulated the problem as a deterministic inventory optimization model in a continuous-time setting and solves the problem using a dynamic programming (DP) approach. Chand *et al.* (1994) considered an SWO problem in a manufacturing environment where some components or parts may be used to substitute for others in an assembly process. Utilizing DP, they derived optimal purchase quantities of the parts for their continuous-time EOQ-based optimization model. Jones *et al.* (1995) formulated a single-period SWO problem as a specially structured plant location problem and solved it using a network flow approach.

Among the limited number of papers on the deterministic models, we observe that most of them are continuous-time EOQ models that consider substitution among two to three products. In this paper, we formulate two variants of the one-way product substitution problem as multi-product DLS models, one for the SWC problem and the other for the SWO problem. We consider very general models with multiple products and with all demands, as well as production costs (fixed and variable), inventory costs, and conversion costs, being time-varying and different across products. In our models, product shortages or backlogging are not allowed. The demand of each product in a period can be satisfied by (i) the production of the product in the same period; (ii) inventories of the product produced in an earlier period; (iii) (in the SWC problem) available stocks converted in the current or earlier periods from lower-index products; or (iv) (in the SWO problem) available stocks of lower-index products in the current period. The objective of our problem is to satisfy the demand of all products at a minimum setup, production, inventory, and (possibly) conversion cost.

Our models are useful to organizations that are seeking opportunities to create greater flexibility in their production and inventory planning by exploring the possibility of product substitution. As pointed out by Chan *et al.* (2002), DLS models such as ours can help a planner make decisions on production and cycle inventory (based on forecasts of demand over a given planning horizon) that balance the trade-offs between fixed production cost and variable costs of production, inventory, and, in our models, conversion. DLS models are particularly useful in situations where demand is non-stationary, which is frequently encountered in production planning, such as Materials Requirements Planning. Further to these planning decisions, the planner typically determines additional safety stock to cope with uncertainties in demand and supply. One clear weakness of our model, as with all deterministic models in the literature, is its inability to address the issue of demand uncertainty. This limits its applicability to some real world situations, such as airline ticket booking, where the main motivation for product substitution is the demand uncertainty and not economies of scale in production and inventory.

Our DLS problems belong to a class of multi-product DLS problems, which are generalizations of the classical single-item Wagner–Whitin DLS problem (Wagner and Whitin 1958). (See Aggarwal and Park (1993) for an extensive review of the classical DLS problem.) Due to the abundance of research in this class, we discuss only a few papers that are related to our work.

Herer and Tzur (2001) studied a discrete-time dynamic transshipment problem, where there are two locations with deterministic demands over a finite planning horizon. The two locations replenish their stocks from a single supplier and transshipments between locations are allowed. This transshipment problem can be viewed as a DLS problem with two products, which can be used to substitute for each other with certain costs. Herer and Tzur developed a DP algorithm to solve a special form of the two-location transshipment problem, where replenishment, inventory holding, and transshipment costs are all stationary (i.e., these costs are constant over the planning horizon).

Lee and Luss (1987) examined a dynamic deterministic capacity expansion problem with multiple facility types. A facility type can be converted to another at a certain conversion cost. Their general model, which includes concave cost functions and allows for backlogging, can be viewed as a multi-product DLS problem with substitution between products. They presented a DP approach that solves a few restricted instances of the general model in polynomial time.

The models that we consider in this paper can also be formulated as special instances of the minimum concave-cost network flow problem (Erickson *et al.* 1987, Guisewite and Pardalos 1993, Lamar 1993, Veinott 1969, and Zangwill 1968). Efficient algorithms have been developed to solve this network flow problem on some special networks, for example, strong-series-parallel networks (Ward 1999) and networks with a fixed number of sources and nonlinear arc costs (Tuy *et al.* 1995). However, the constructed networks that are equivalent to our models (see discussions of network construction in Sections 2 and 3) do not belong to any of these special networks.

The rest of the paper is organized as follows. In Section 2, we present the model for the SWC problem. We discuss its computational complexity as well as some properties of its optimal solution. A DP algorithm is then developed to solve the problem. In Section 3, we discuss another model for the SWO problem. We show the relationship between the first and second models and develop a more efficient algorithm to solve the second model. In Section 4, we develop a heuristic algorithm for solving large-sized problems. Computational experiments are conducted to test the efficiency and effectiveness of our algorithms. We conclude the paper in Section 5.

2. One-way Substitution with Conversion

2.1. Problem Formulation

In this section, we consider the SWC problem with m products denoted by $P_1, P_2, \ldots,$ P_m, where for any j and k $(1 \leq j < k \leq m)$, P_j can be converted to P_k with a conversion cost that depends on the time the conversion occurs. In each time period, the following sequence of events is assumed: (i) production and delivery of all products, (ii) conversion of products, and (iii) demand of all products. In step (ii), the conversion of P_1 to other products takes place first, followed by the conversion of P_2 to other products, followed in turn by the conversion of P_3 to other products, and so on.

The following notation is used in our model:

m = total number of different products;

T = total number of time periods in the planning horizon;

D_{tj} = demand of P_j in period t, where $1 \leq t \leq T$ and $1 \leq j \leq m$;

K_{tj} = setup cost of the production of P_j in period t, where $1 \leq t \leq T$ and $1 \leq j \leq m$;

p_{tj} = unit production cost of P_j in period t, where $1 \leq t \leq T$ and $1 \leq j \leq m$;

h_{tj} = unit holding cost of P_j from period t to period $t+1$, where $1 \leq t \leq T-1$ and $1 \leq j \leq m$;

c_{tjk} = unit conversion cost from P_j to P_k in period t, where $1 \leq t \leq T$ and $1 \leq j < k \leq m$.

The conversion cost in our model is applicable to general situations where the conversion process uses resources such as utility, labor, and equipment, and costs may vary from one period to another. We remark that in some real world applications, the conversion cost may not differ from period to period (for example, simple re-packaging in the retailer example cited in Section 1). However, even under the assumption of stationary conversion costs, i.e., $c_{tjk} = c_{jk}$ for all t, theoretically the SWC and SWO problems are not identical. One reason for this is the fact that with conversion in the SWC problem and time-varying inventory holding costs for different products, a decision maker has the option of converting a lower-index product and not using the converted product immediately for substitution. This option is not available in the SWO problem. However, if we further assume that in any period t, the inventory holding cost for a lower-index product i is no larger than that of a higher-index product j, i.e., $h_{ti} \leq h_{tj}$ for $i < j$, then the instance of the SWC problem becomes a special case of the SWO problem. The reason is that, in this case, the decision maker has no incentive to convert a product, which has a lower inventory holding cost, before it is needed to substitute for another product with a higher inventory holding cost.

All of the above parameters are assumed to be nonnegative, and a cost parameter may be equal to $+\infty$. The decision variables of the problem are:

x_{tj} = number of units of product P_j produced in period t;

I_{tj} = number of units of product P_j held in inventory from period t to period $t+1$;

y_{tjk} = number of units of product P_j converted into product P_k in period t.

Define

$$\delta(x) = \begin{cases} 1, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}$$

Assuming a zero inventory for all products at the beginning of period 1, the SWC problem can be formulated as the following mathematical program:

$$\begin{aligned}
\text{minimize} \quad & \sum_{t=1}^{T}\sum_{j=1}^{m}\Big[K_{tj}\delta(x_{tj}) + p_{tj}x_{tj}\Big] + \sum_{t=1}^{T-1}\sum_{j=1}^{m} h_{tj}I_{tj} + \sum_{t=1}^{T}\sum_{j=1}^{m-1}\sum_{k=j+1}^{m} c_{tjk}y_{tjk} \\
\text{subject to} \quad & I_{tj} = I_{t-1,j} + x_{tj} + \sum_{k=1}^{j-1} y_{tkj} - \sum_{k=j+1}^{m} y_{tjk} - D_{tj} \quad (1 \le t \le T;\ 1 \le j \le m) \\
& I_{0j} = 0 \quad (1 \le j \le m) \\
& I_{tj}, x_{tj} \ge 0 \quad (1 \le t \le T;\ 1 \le j \le m) \\
& y_{tjk} \ge 0 \quad (1 \le t \le T;\ 1 \le j < k \le m).
\end{aligned}$$

We denote this problem as **SWCP**.

Note that for simplicity, we have assumed that there is no fixed cost associated with each conversion in **SWCP**. Thus, this model is applicable to situations where the effort of setting up the conversion process is minimal, for example, when conversion involves only breaking larger packages of a product to substitute for a shortage of the same product packaged in smaller sizes. For other applications, it is more reasonable to include fixed conversion costs. In Section 5, we will discuss how our solution method can be extended to handle such situations.

2.2. Some Properties of SWCP

The following theorem implies that the existence of a polynomial time algorithm for this problem is unlikely.

Theorem 1. SWCP is NP-hard in the strong sense.

The proof of this theorem will be made apparent in the next section, when we establish that the SWO problem studied in that section is strongly NP-hard and that it can be transformed in polynomial time into a special instance of **SWCP**.

In the next subsection, we will develop a DP algorithm to solve **SWCP**, which runs in polynomial time with a fixed number of products m. For ease of exposition, we assume that all demands are strictly positive in our algorithm development. The inclusion of the case where some demands can be zero will not change the structure of our solution methods, but will make our argument a little more tedious.

We now represent **SWCP** as a minimum concave cost network flow problem on a specially constructed network G with a single source and multiple destination nodes (see Figure 1). The source node is denoted as v_0. There are mT destination nodes, which can be categorized into m levels. The t^{th} node at the j^{th} level is denoted as v_{tj}. Associated with each destination node v_{tj} is a demand quantity D_{tj}. The arcs in this network can be categorized into three groups. The first group consists of "production" arcs $v_0 \to v_{tj}$, where $t = 1, 2, \ldots, T$ and $j = 1, 2, \ldots, m$. A flow x_{tj} on arc $v_0 \to v_{tj}$ represents a setup and production of product P_j in period t. The second group consists of "inventory" arcs $v_{tj} \to v_{t+1,j}$, where $t = 1, 2, \ldots, T-1$ and $j = 1, 2, \ldots, m$. A flow I_{tj} on arc $v_{tj} \to v_{t+1,j}$ represents a holding of inventory of P_j from period t into the next period. The third group consists of "conversion" arcs $v_{tj} \to v_{tk}$, where $t = 1, 2, \ldots, T$ and $1 \leq j < k \leq m$. A flow

y_{tjk} on arc $v_{tj} \rightarrow v_{tk}$ represents a conversion from product P_j to product P_k occurred in period t.

The objective of the network flow problem, denoted as **NF**, is to find a feasible flow in this network such that the total cost is minimized, while the demand associated with each destination node is satisfied. It is easy to show the equivalence of **SWCP** and **NF**. Note that **NF** is a single-source minimum concave cost network flow problem, and that we have the following important property.

Property 1. There exists an optimal solution to a single-source minimum concave cost network flow problem with at most one positive incoming flow into each node.
Proof: See Zangwill (1968).

Applying Property 1 to problem **NF** and to **SWCP**, we have the next property.

Property 2. There exists an optimal solution to **SWCP** such that for every $t \in [1, T]$ and $j \in [1, m]$, the D_{tj} units of demand of product j in period t are *all* satisfied by the production of a single product P_π that takes place in a single period τ, plus a series of conversions and inventory holdings (if necessary), for some $\pi \leq j$ and $\tau \leq t$.

Property 2 implies that it is optimal to substitute the demand of a product in a certain period with another *single* product. This property holds in our model because of the assumptions that demands are deterministic, all cost functions are concave (the cost functions in our model are all special forms of concave functions), and production and conversion capacities are unlimited. In other situations with capacity restrictions or stochastic demands, it may be desirable to use a combination of two or more products to substitute for the demand of a single product.

Suppose that the demand of P_j in period t is satisfied by the production of $P_{\pi_{tj}}$ that takes place in period τ_{tj}, where $1 \leq \tau_{tj} \leq t \leq T$ and $1 \leq \pi_{tj} \leq j \leq m$ (see Property 2). The following property characterizes the relationship among (τ_{t1}, π_{t1}), ...,

(τ_{tm}, π_{tm}), $(\tau_{t+1,1}, \pi_{t+1,1})$, …, $(\tau_{t+1,m}, \pi_{t+1,m})$.

Property 3. There exists an optimal solution to **SWCP** that satisfies Properties 1 and 2 and the following condition: For $j = 1, 2, \ldots, m$,

$$(\tau_{t+1,j}, \pi_{t+1,j}) \in \Phi_j,$$

where $\Phi_j = \big\{(\tau_{tj}, \pi_{tj}), (\tau_{t+1,1}, \pi_{t+1,1}), \ldots, (\tau_{t+1,j-1}, \pi_{t+1,j-1}), (t+1, j)\big\}$.

Proof. By Property 1, there is at most one positive incoming flow into node $v_{t+1,j}$. There are three possible cases:

- Case 1: The incoming flow at node $v_{t+1,j}$ comes from node v_{tj}. In this case, the demand of P_j in period $t+1$ is satisfied by the same production run as that of P_j in period t, and therefore, $(\tau_{t+1,j}, \pi_{t+1,j}) = (\tau_{tj}, \pi_{tj})$.
- Case 2: The incoming flow at node $v_{t+1,j}$ comes from node $v_{t+1,k}$, for some $k = 1, 2, \ldots, j–1$. In this case, the demand of P_j in period $t+1$ is satisfied by the same production run as that of P_k in the same period, and therefore, $(\tau_{t+1,j}, \pi_{t+1,j}) = (\tau_{t+1,k}, \pi_{t+1,k})$.
- Case 3: The incoming flow at node $v_{t+1,j}$ comes directly from v_0. In this case, $(\tau_{t+1,j}, \pi_{t+1,j}) = (t+1, j)$.

Combining Cases 1, 2, and 3 gives us the desired result. ∎

2.3. DP Algorithm for Solving SWCP

With Properties 1, 2, and 3, we now develop a backward DP algorithm for solving **SWCP**. Note that all of the costs except for the setup cost in our model are linear with the number of units involved. Thus, the costs involved include a fixed setup cost in period τ to produce P_π and a *variable cost* to satisfy the demand of P_j in period t with the production of P_π in period τ. This variable cost consists of the production cost, conversion cost (if any), and inventory holding cost (if any).

Define

$$\Gamma_{tj} = \big\{(\tau, \pi) \bigm| \tau = 1, 2, \ldots, t;\ \pi = 1, 2, \ldots, j\big\},$$

which is the set of all time–product combinations that could be used to satisfy the demand of P_j in period t. For $1 \leq t \leq T$ and $(\tau_{tj}, \pi_{tj}) \in \Gamma_{tj}$ $(j = 1, 2, \ldots, m)$, we denote $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ as a restricted version of **SWCP** that satisfies the following conditions:

(i) The demand of all products in periods t through T is met by production in periods 1 through T.

(ii) The demand of P_j $(1 \leq j \leq m)$ in period t is satisfied by the production of product $P_{\pi_{tj}}$ that takes place in period τ_{tj}.

(iii) The objective of the problem is to find a solution that satisfies Properties 1, 2, and 3, and minimizes the total variable cost plus all the setup costs incurred in periods t through T to meet all demands in periods t through T.

We define the following quantities:

- $F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ = minimum objective value for problem $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$;
- $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ = total variable cost needed to satisfy the demands of $P_1, P_2, \ldots, P_m$ in period t in an optimal solution of $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$;
- $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ = total setup cost incurred in period t in an optimal solution of $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$.

It is noteworthy that the optimal value function F_t includes not only all costs incurred in periods $t, t+1, \ldots, T$, but also part of the variable cost incurred before period t. It is this newly designed cost-counting scheme that leads to the following DP formulation, which can be solved in polynomial time when m is fixed.

By the above definition, the optimal objective value of **SWCP** is given by

$$\min \left\{ F_1(1, 1, \ldots, 1; \pi_{11}, \pi_{12}, \ldots, \pi_{1m}) \;\middle|\; \pi_{1k} \in \{1, 2, \ldots, k\};\ k = 1, 2, \ldots, m \right\}.$$

We now develop a recurrence relation to calculate $F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$, for $1 \leq t \leq T$. Suppose that in an optimal solution to problem

$SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$, the demand of P_k $(1 \le k \le m)$ in period $t+1$ is satisfied by the production of $P_{\pi_{t+1,k}}$ that took place in period $\tau_{t+1,k}$. By Property 3, we have the following recurrence relation:

$$\begin{aligned}
&F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}) \\
&\quad = f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}) + s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}) \\
&\qquad + \min\Big\{ F_{t+1}(\tau_{t+1,1}, \tau_{t+1,2}, \ldots, \tau_{t+1,m}; \pi_{t+1,1}, \pi_{t+1,2}, \ldots, \pi_{t+1,m}) \\
&\qquad\qquad\qquad \Big|\ (\tau_{t+1,k}, \pi_{t+1,k}) \in \Phi_k;\ k = 1, 2, \ldots, m \Big\}, \qquad (1)
\end{aligned}$$

for $1 \le t \le T$ and $(\tau_{tj}, \pi_{tj}) \in \Gamma_{tj}$ $(j = 1, 2, \ldots, m)$, where Φ_k is defined in Property 3. The boundary conditions are:

$$F_{T+1}(\tau_{T+1,1}, \tau_{T+1,2}, \ldots, \tau_{T+1,m}; \pi_{T+1,1}, \pi_{T+1,2}, \ldots, \pi_{T+1,m}) = 0,$$

for $(\tau_{T+1,j}, \pi_{T+1,j}) \in \Gamma_{T+1,j}$ $(j = 1, 2, \ldots, m)$.

We now turn our attention to the computation of the values of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ and $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$. Define $g_{tj}(\tau, \pi)$ as the minimum variable cost needed to satisfy a unit demand of P_j in period t by the production of product P_π that took place in period τ, for $1 \le \tau \le t \le T$ and $1 \le \pi \le j \le m$. It is easy to see that $g_{tj}(\tau, \pi)$ is equal to the production cost $p_{\tau\pi}$ plus the shortest "distance" from node $v_{\tau\pi}$ to node v_{tj} in network G if we view the unit cost of an arc as its arc length. Hence, all $g_{tj}(\tau, \pi)$ values can be determined by solving an all-pairs shortest path problem, which requires a running time of $O\big((mT)^3\big)$ (see p. 156 of Ahuja *et al.* 1993). In fact, due to the special structure of network G, the values of $g_{tj}(\tau, \pi)$ can also be determined in $O(m^3T^2)$ time using a recursive procedure. However, this will not affect the overall complexity of our algorithm.

With the definition of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$, we clearly have

$$f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}) = \sum_{k=1}^{m} D_{tk} g_{tk}(\tau_{tk}, \pi_{tk}).$$

Since $1 \leq \tau_{tj} \leq T$ and $1 \leq \pi_{tj} \leq j$ for $j = 1, 2, \ldots, m$, the number of combinations of $t, \tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}, \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}$ is $O(m!T^{m+1})$. This implies that, given all the values of $g_{tj}(\tau, \pi)$, we can determine all the values of $\big\{f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})\big\}$ in $O\big((m+1)!T^{m+1}\big)$ time.

Note that the value of $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ can be determined without knowing the optimal solution of $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$. Suppose that in an optimal solution to $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$, the demand of P_j $(1 \leq j \leq m)$ in period τ $(t \leq \tau \leq T)$ is satisfied by the production of P_π that took place in period t. We must then have a setup in period t to produce product P_π. By Property 1, the demand of P_π in period t must be satisfied by its own production in the same period. By definition of $\tau_{t\pi}$, we have $\tau_{t\pi} = t$. From the above discussion, we see that

$$s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}) = \sum_{\pi \in \{\pi_{tk} \,|\, \tau_{tk} = t;\, k=1,\ldots,m\}} K_{t\pi},$$

for $t = 1, 2, \ldots, T$ and $(\tau_{tj}, \pi_{tj}) \in \Gamma_{tj}$ $(j = 1, 2, \ldots, m)$. This equation implies that all of the values of $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ can be obtained in $O\big((m+1)!T^{m+1}\big)$ time as well.

Finally, note that the number of elements in the set $\big\{(\tau_{t+1,1}, \tau_{t+1,2}, \ldots, \tau_{t+1,m}; \pi_{t+1,1}, \pi_{t+1,2}, \ldots, \pi_{t+1,m}) \;\big|\; (\tau_{t+1,k}, \pi_{t+1,k}) \in \Phi_k;\; k = 1, 2, \ldots, m\big\}$ is at most $(m+1)!$. Hence, evaluating the minimization on the right hand side of equation (1) takes $O\big((m+1)!\big)$ time. As mentioned earlier, the number of combinations of $t, \tau_1, \tau_2, \ldots, \tau_m, \pi_1, \pi_2, \ldots, \pi_m$ is $O(m!T^{m+1})$. Hence, the effort to compute all values of $\big\{F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})\big\}$ takes $O\big((m+1)!m!T^{m+1}\big)$ time, provided that all of the values of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ and $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ are available. Together with the preprocessing efforts discussed earlier, we see that the overall complexity of our algorithm is $O\big((m+1)!m!T^{m+1}\big)$. This implies that when the number of products m is fixed, the running time of our algorithm is $O(T^{m+1})$.

3. One-way Substitution without Conversion

3.1. Problem Formulation

In this section, we consider the SWO problem (denoted as **SWOP**) with m products $P_1, P_2, \ldots, P_m$. In this problem, these m products represent m different grades of a single product, where product P_1 has the highest grade, P_2 has the second-highest grade, and so on. We assume that a direct downward substitution from a higher-grade product to a lower-grade product is always allowed and that no physical transformation is incurred when the substitution takes place. That is, if $1 \leq j < k \leq m$, then P_j can substitute P_k without incurring any conversion cost. This type of substitution may occur in the situation where a customer's order for some lower-grade product arrives when such product is out of stock. The supplier may choose to use the inventory of some higher-grade product to fulfill the order and charge the price of the lower-grade product. This can avoid the setup cost of directly producing the lower-grade product. However, the production cost and holding cost for the higher-grade product are usually higher than those for the lower-grade product (although we do not make this assumption in our model). Thus, the challenge to the supplier is to plan the replenishment of all products so as to minimize the total setup, production, and inventory holding costs.

In **SWOP**, there are T periods in the planning horizon. In each time period t, the demand of product P_k can be satisfied either by the inventory of P_k at the end of period $t-1$, or by the direct substitution with a higher-grade product P_j (where $j < k$), or by the new setup and production of P_k in the same period t. Recall that in **SWCP**, if a substitution is necessary, a lower-index product could be converted and held as the inventory of another higher-index product before it is used to meet the demand of the latter. On the contrary, no conversion is needed in **SWOP**. Thus, wherever a substitution takes place, a higher-grade product is always held in its own inventory (if an inventory is necessary) until the moment when it is used to meet the demand of another lower-grade product.

In the following discussion, we will first demonstrate that **SWOP** can actually be formulated as a special case of **SWCP**. This implies that the DP algorithm developed in Section 2 can be used to solve the problem. However, in this section we will develop a more efficient algorithm for this special case.

We reuse the notation m, T, D_{tj}, K_{tj}, p_{tj}, h_{tj}, x_{tj}, I_{tj}, and y_{tjk} as defined in Section 2. Given any instance of **SWOP** with parameters m, T, D_{tj}, K_{tj}, p_{tj}, and h_{tj}, we construct the corresponding instance of **SWCP** with $2m$ "products" and T periods, where its parameters m', T', D'_{tj}, K'_{tj}, p'_{tj}, h'_{tj}, and c'_{tjk} are defined as follows:

$$m' = 2m;$$

$$T' = T;$$

$$D'_{tj} = \begin{cases} 0, & \text{if } j \text{ is odd,} \\ D_{t,j/2}, & \text{if } j \text{ is even;} \end{cases}$$

$$K'_{tj} = \begin{cases} K_{t,(j+1)/2}, & \text{if } j \text{ is odd,} \\ +\infty, & \text{if } j \text{ is even;} \end{cases}$$

$$p'_{tj} = \begin{cases} p_{t,(j+1)/2}, & \text{if } j \text{ is odd,} \\ +\infty, & \text{if } j \text{ is even;} \end{cases}$$

$$h'_{tj} = \begin{cases} h_{t,(j+1)/2}, & \text{if } j \text{ is odd,} \\ +\infty, & \text{if } j \text{ is even;} \end{cases}$$

$$c'_{tjk} = \begin{cases} 0, & \text{if } j \text{ is odd, } k \text{ is even, and } k > j, \\ +\infty, & \text{otherwise.} \end{cases}$$

Note that in this construction, each product of **SWOP** is broken into two. This includes a product (odd level product) that inherits the production costs of the original but has no demand, and a second product (even level product) that inherits the demand and substitution pattern of the original product but is prohibitively expensive to produce. The network for the above constructed instance of **SWCP** is depicted in Figure 2, in which the arcs with infinite cost are not shown.

In this network, there is no cost of conversion. However, once a product gets converted into a higher-indexed product, it will arrive at an even level. Once it arrives at such a level, it cannot be held in inventory (since there is no horizontal arc at an even level). This satisfies the requirement of **SWOP**, where the original product is always held in its own inventory until the moment when it is used to meet the demand of a higher-index product.

3.2. Some Properties of SWOP

We have shown that **SWOP** is a special case of **SWCP**, and the transformation from **SWOP** to **SWCP** can be done in polynomial time. The next theorem states that this special case remains strongly NP-hard when m is not fixed. The proof of this theorem also serves as the proof for Theorem 1 in Section 2.

Theorem 2. SWOP is NP-hard in the strong sense.

Proof: See Appendix.

If we formulate **SWOP** as an instance of **SWCP** using the above transformation and then directly apply the algorithm developed in Section 2 to solve it, the computational complexity will be $O\big((2m+1)!(2m)!T^{2m+1}\big)$. This complexity becomes $O(T^{2m+1})$ when m is fixed. However, by exploring the special structure of **SWOP**, we will develop a more efficient DP algorithm to solve **SWOP** directly.

Given an arbitrary period t $(1 \leq t \leq T)$, define the *latest production period* of a product P_k $(1 \leq k \leq m)$ in periods 1 through t as the largest indexed period τ with $x_{\tau k} > 0$ and $\tau \leq t$ ($\tau = 0$ if such a period does not exist). Suppose some demands of a product P_j in a period t are satisfied by a stock of product P_k $(k \leq j)$ in period t, which is produced in period τ, where $1 \leq \tau \leq t$. We define the *variable cost* of satisfying the per unit demand of P_j in period t by the production of P_k in period τ as the unit variable production cost of P_k in period τ plus the total inventory costs to hold each unit of P_k from period τ to period t. We are now ready to present a property of the optimal solution to **SWOP**.

Property 4. There exists an optimal solution to **SWOP** where (i) each demand D_{tj} ($1 \le t \le T$; $1 \le j \le m$) is satisfied *entirely* either by the latest production of product P_j in time periods 1 through t, or by the latest production of another lower-index product P_k ($k < j$) in periods 1 through t; (ii) in the latter case, the same production of P_k also satisfies the demand $D_{t,j-1}$.

Proof: Condition (i) is the direct consequence of applying Property 1 to the minimum concave cost network flow problem on the constructed network in Figure 2, which is equivalent to **SWOP**. To prove (ii), suppose in an optimal solution that satisfies condition (i), demand D_{tj} is fulfilled by some lower-index product P_k ($k < j$). Also suppose that in the same solution, demand $D_{t,j-1}$ is satisfied by the latest production of product P_ℓ, where $\ell \le j-1$. If $k \ne \ell$, we can modify the solution by comparing the variable costs associated with products P_k and P_ℓ, and selecting the cheaper one to satisfy both demands $D_{t,j-1}$ and D_{tj}. It is easy to see that this modification will not increase the total cost of the solution and will not result in a solution that violates condition (i). By repeatedly applying this modification, we will obtain an optimal solution that satisfies conditions (i) and (ii). ▮

3.3. DP Algorithm for Solving SWOP

In the following analysis, we only consider solutions that satisfy Property 4. For $1 \le t \le T$ and $0 \le \tau_{tj} \le t$ ($j = 1, 2, \ldots, m$), we denote $SWOP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ as a restricted version of SWOP that satisfies the following conditions:

(i) The demand of all products in periods t through T is met by production in periods 1 through T.

(ii) The latest production of P_j ($1 \le j \le m$) in periods 1 through t takes place in period τ_{tj}.

(iii) The objective of the problem is to find a solution that satisfies Property 4 and minimizes the total variable cost plus all of the setup costs incurred in periods t through T to meet all demand in periods t through T.

We define the following quantities:

- $F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ = minimum objective value for problem $SWOP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$;
- $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ = total variable cost needed to satisfy the demand of $P_1, P_2, \ldots, P_m$ in period t in an optimal solution of $SWOP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$;
- $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ = total setup cost incurred in period t in an optimal solution of $SWOP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$.

Clearly, the optimal objective value of **SWOP** is given by

$$\min\left\{F_1(\tau_{11}, \tau_{12}, \ldots, \tau_{1m}) \;\middle|\; \tau_{11} = 1 \text{ and } \tau_{1j} \in \{0,1\};\ j = 2, 3, \ldots, m\right\}.$$

We now develop a recurrence relation to calculate $F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$, for $1 \le t \le T$. We note that for each product P_k ($k = 1, 2, \ldots, m$), if the latest production period in periods 1 through t is τ_{tk}, then the latest production period $\tau_{t+1,k}$ in periods 1 through $t+1$ is either τ_{tk} or $t+1$. Thus, we have the following recurrence relation:

$$\begin{aligned} F_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}) = {} & f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}) + s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}) \\ & + \min\Big\{F_{t+1}(\tau_{t+1,1}, \tau_{t+1,2}, \ldots, \tau_{t+1,m}) \\ & \qquad \Big|\ \tau_{t+1,k} = t{+}1 \text{ or } \tau_{tk}, \text{ for } k = 1, 2, \ldots, m\Big\}, \end{aligned} \tag{2}$$

for $1 \le t \le T$ and $1 \le \tau_{tj} \le t$ ($j = 1, 2, \ldots, m$). The boundary conditions are:

$$F_{T+1}(\tau_{T+1,1}, \tau_{T+1,2}, \ldots, \tau_{T+1,m}) = 0,$$

for $1 \le \tau_{T+1,j} \le T+1$ ($j = 1, 2, \ldots, m$).

To compute the values of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ and $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$, we first define $g_{tj}(\tau)$ as the minimum variable cost needed to satisfy unit demand of P_j in period t by the production of the same product P_j that took place in period τ, for $1 \le \tau \le t \le T$ and $j = 1, 2, \ldots, m$. It is easy to see that

$$g_{tj}(\tau) = \begin{cases} p_{\tau j}, & \text{if } t = \tau, \\ g_{t-1,j}(\tau) + h_{t-1,j}, & \text{if } t > \tau, \end{cases}$$

for $1 \leq \tau \leq t \leq T$ and $j = 1, 2, \ldots, m$. The number of combinations of t, j, and τ is $O(mT^2)$. Thus, all $g_{tj}(\tau)$ values can be determined in $O(mT^2)$ time.

Given an optimal solution to problem $SWOP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$, let z_{tk} be the minimum variable cost of satisfying unit demand of P_k in period t. From Property 4, we know that demand D_{tk} is satisfied either by the production of P_k that took place in period τ_{tk}, or by the production that satisfies the demand $D_{t,k-1}$. Thus, $z_{t1} = g_{t1}(\tau_1)$, and for $k = 2, 3, \ldots, m$, we have $z_{tk} = \min\{g_{tk}(\tau_k), z_{t,k-1}\}$. For a given t, with all $z_{t1}, z_{t2}, \ldots, z_{tm}$ computed in $O(m)$ time, we have

$$f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}) = \sum_{k=1}^{m} D_{tk} z_{tk}.$$

The number of combinations of $t, \tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}$ is $O(T^{m+1})$. For each combination of $t, \tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}$, obtaining the value of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ requires $O(m)$ time. Hence, after predetermining the values of $g_{tj}(\tau)$, all of the values of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ can be obtained in $O(mT^{m+1})$ time.

With an argument similar to that for computing $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$ in the Section 2, we have

$$s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}) = \sum_{\ell \in \{k | \tau_{tk} = t;\, k=1,\ldots,m\}} K_{t\ell},$$

for $t = 1, 2, \ldots, T$ and $1 \leq \tau_{tj} \leq t$ $(j = 1, 2, \ldots, m)$. Therefore, all of the values of $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ can also be determined in $O(mT^{m+1})$ time.

We see now that predetermining all of the values of $f_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ and $s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm})$ requires $O(mT^{m+1})$ time. The number of combinations of $t, \tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}$ is $O(T^{m+1})$. For each combination, evaluating the minimization on the right hand side of equation (2) takes $O(2^m)$ time. Hence, the overall complexity of this algorithm is $O(2^m T^{m+1})$. This implies that when m is fixed, the running time of our algorithm is $O(T^{m+1})$.

4. Heuristic Method and Computational Experiments

In this section, we develop a heuristic for **SWCP** and perform a numerical study, in which its effectiveness is tested and a sensitivity analysis of the model parameters is provided. In addition, we test the efficiency of our DP algorithm developed earlier. Since **SWOP** is a special case of **SWCP**, the same heuristic can be applied to solve **SWOP**. Therefore, for simplicity, our numerical study is performed only on **SWCP**.

Our heuristic is an extension of the well known Silver–Meal heuristic (Silver and Meal 1973). In this heuristic, we first rearrange the nodes $\{v_{tj} \mid t = 1, 2, \ldots, T;\ j = 1, 2, \ldots, m\}$ of network G into a one-dimensional array $\{\tilde{v}(n) \mid n = 1, 2, \ldots, mT\}$ such that $v_{tj} = \tilde{v}((j-1)T + t)$. In other words, using the drawing depicted in Figure 1, the nodes in the first row of network G are placed in front of the nodes in the second row, which are placed in front of the nodes in the third row, and so on. At each node $\tilde{v}(n)$, a set of four elements $\{S(n), U(n), C(n), N(n)\}$ is stored. Element $S(n)$ is the "setup node" of $\tilde{v}(n)$, which is the node where a setup is made to fulfill the demand of the associated product j at $\tilde{v}(n)$. Element $U(n)$ is the total variable cost (including production, inventory holding, and conversion costs) of satisfying one unit of demand at $\tilde{v}(n)$ from its setup node $S(n)$. Element $C(n)$ is the total cost (including setup and variable costs) of satisfying the demands at $\tilde{v}(n)$, the predecessor of $\tilde{v}(n)$, the predecessor of the predecessor of $\tilde{v}(n)$, etc., from the setup node $S(n)$. Element $N(n)$ is the number of nodes along the path from the setup node $S(n)$ to node $\tilde{v}(n)$. Thus, $C(n)/N(n)$ is a counterpart of the quantity that represents the average cost per period from the latest setup node up to node $\tilde{v}(n)$ in the Silver–Meal heuristic in the case of a single product. The heuristic then proceeds to sequentially assign values to $\{S(n), U(n), C(n), N(n)\}$ for $n = 1, 2, \ldots, mT$ in the following way. In contrast to the single product case, the incoming flow to node $\tilde{v}(n) = \tilde{v}((j-1)T + t)$ can emanate from node $\tilde{v}((j-1)T + t - 1)$ (when $t > 1$) or nodes $\tilde{v}(t), \tilde{v}(T+t), \tilde{v}(2T+t), \ldots, \tilde{v}((j-2)T+t)$. Of these nodes, it is natural to select the one (denoted by $\tilde{v}(n')$) with the minimum total variable cost of satisfying one unit of demand at node $\tilde{v}(n)$. Then similar to the Silver–

Meal heuristic, a decision on $S(n)$ is made at node $\tilde{v}(n)$ by comparing $C(n')/N(n')$ with $[C(n') + U(n)D(n)]/[N(n') + 1]$, where $D(n)$ is the demand at $\tilde{v}(n)$.

Define $\phi(n)$ as the set of all possible immediate predecessors of $\tilde{v}(n)$ in network G (for example, $\phi(2) = \{1\}$ and $\phi(T + 2) = \{T+1, 2\}$). Denote $K(n)$ and $p(n)$ as the setup cost and unit production cost, respectively, at $\tilde{v}(n)$. Define $\eta(n, n')$ as the unit cost from $\tilde{v}(n)$ to $\tilde{v}(n')$, where $\tilde{v}(n)$ is an immediate predecessor of $\tilde{v}(n')$ in network G (for example, $\eta(1, 2) = h_{11}$ and $\eta(1, T+1) = c_{112}$). A formal description of this "extended Silver–Meal heuristic" is given below.

Heuristic H for SWCP:

Step 1: (Initialization.) Set $S(1) = \tilde{v}(1)$, $U(1) = p(1)$, $C(1) = K(1) + p(1)D(1)$, and $N(1) = 1$.

Step 2: For $n = 2, 3, \ldots, mT$, set $n' \leftarrow \arg\min_{x\in\phi(n)}\{U(x) + \eta(x, n)\}$, i.e., $\tilde{v}(n')$ is the immediate predecessor of $\tilde{v}(n)$ with the minimum variable cost of satisfying one unit of demand at node $\tilde{v}(n)$. If $C(n')/N(n') > [C(n') + (U(n') + \eta(n', n))D(n)]/[N(n') + 1]$, then let node $\tilde{v}(n)$ share the same setup as node $\tilde{v}(n')$, that is, set $S(n) = S(n')$, $U(n) = U(n')+\eta(n', n)$, $C(n) = C(n')+U(n)D(n)$, and $N(n) = N(n')+1$. Otherwise, create a new setup at node $\tilde{v}(n)$, that is, set $S(n) = \tilde{v}(n)$, $U(n) = p(n)$, $C(n) = K(n) + p(n)D(n)$, and $N(n) = 1$.

Step 3: (Computing the total cost.) Let Φ be the set of setup nodes. Thus, $\Phi = \{S(n) \mid n = 1, 2, \ldots, mT\}$. Then, the total cost = (total setup cost) + (total variable cost) = $\sum_{\tilde{v}(n)\in\Phi} K(n) + \sum_{n=1}^{mT} U(n)D(n)$.

It is easy to check that the time complexity of Heuristic H is $O(m^2T)$. Next, we discuss the parameter setting of the computational study and present the numerical results to demonstrate the effectiveness of Heuristic H. In this study, we also compute the performance of the Wagner–Whitin algorithm (Wagner and Whitin 1958) when applied to the m products independently without considering product substitution. Let Z^H denote the total cost of the solution generated by Heuristic H. Let Z^W denote the total cost of the solution generated by the Wagner–Whitin algorithm when applied to the m products independently. Let Z^* denote the total cost of the optimal solution obtained by the DP algorithm developed in Section 2. Let $R^H = (Z^H - Z^*)/Z^* \times 100\%$ be the relative error of the solution generated by Heuristic H. Let $R^W = (Z^W - Z^*)/Z^* \times 100\%$ be the relative error of the solution if product substitution is ignored.

The cost parameters include the setup cost K_{tj}, production cost p_{tj}, holding cost h_{tj}, and conversion cost c_{tjk}, where $1 \le t \le T$ and $1 \le j < k \le m$. Another parameter is the demand D_{tj}. The variability of data can be categorized into two types: The variability of each product over the time horizon and the variability across different products. We make a few assumptions on the parameters K_{tj} and h_{tj}. First, we assume that the holding cost is proportional to the production cost (this assumption is realistic when the holding cost is proportional to the monetary investment). We set $h_{tj} = \lambda p_{tj}$ for all t and j. We call λ the "holding cost factor." Second, we set $K_{tj} = \bar{K}_j$, for all $t = 1, 2, \ldots, T$, that is, the setup cost is stationary. The setup cost is chosen in such a way that an EOQ order cycle of length 2 is obtained, that is, $\sqrt{2\bar{K}_j/\bar{h}_j\bar{D}_j} = 2$, or equivalently, $\bar{K}_j = 2\bar{h}_j\bar{D}_j$, where $\bar{h}_j$ and $\bar{D}_j$ are the average holding cost rate and the average demand over the time horizon for product j, respectively. Third, the conversion cost c_{tjk} consists of the differential of the product costs, $\max\{p_{tk} - p_{tj}, 0\}$, and a "repackaging fee" of μp_{tj}. We call μ the "conversion cost factor." Thus, based on the above assumptions, m, T, λ, μ, p_{tj}, and D_{tj} determine all of the input parameters of our computational study.

The computational experiments are carried out in two main parts. The first part is to test the effectiveness of Heuristic H and to show the stability of the results with respect to m and T. For these purposes, we let α_{time}, α_{prdt}, β_{time}, and β_{prdt} be random values selected from $\{0, 0.1, 0.2\}$, λ be a random value selected from $\{0.25, 0.5, 1\}$, and μ be a random value selected from $\{0, 0.1, 0.2\}$. Then, for $j = 1, 2, \ldots, m$, we generate a value for $\bar{D}_j$ that is normally distributed with a mean of 1 and a standard deviation α_{prdt} (we set $\bar{D}_j = 0$ if the value generated is negative). Similarly, for $j = 1, 2, \ldots, m$, we generate a value for $\bar{p}_j$ that is normally distributed with a mean of 1 and a standard deviation β_{prdt}. Then, for $t = 1, 2, \ldots, T$, we generate a value for p_{tj} that is normally distributed with mean $\bar{p}_j$ and standard deviation $\beta_{\text{time}}\bar{p}_j$, and we generate a value for D_{tj} that is normally distributed with mean $\bar{D}_j$ and standard deviation $\alpha_{\text{time}}\bar{D}_j$. Thus, α_{time} and α_{prdt} represent the demand variability over the time horizon and across different products, respectively, while β_{time} and β_{prdt} represent the production cost variability over the time horizon and across different products, respectively. We randomly select 30 different sets of values for $(\alpha_{\text{time}}, \beta_{\text{time}}, \alpha_{\text{prdt}}, \beta_{\text{prdt}}, \lambda, \mu)$. For each of them, we repeat the generation of p_{tj} and D_{tj} ten times. Thus, we test 300 instances for each pair of (m, T). We compute the average values of R^H and R^W over all of these 300 instances for the cases of $(m = 2, T = 10)$, $(m=2, T=20)$, $(m=2, T=40)$, $(m=2, T=80)$, $(m=3, T=10)$, $(m=3, T=20)$, and $(m=3, T=40)$ using a 3 GHz processor with 1 GB of RAM. Note that in many real world applications, the number of products that are hierarchically substitutable is quite small. Thus, in this computational study, we focus on test instances with two and three products. The results are summarized in Table 1, from which we observe that R^H is quite stable with respect to both m and T, although we do observe a small trend of increasing R^H with respect to m. In contrast, R^W is stable with respect to T, but unstable with respect to m. Hence, the benefit of product substitution increases significantly as the number of products increases. This is plausible, since adding more products enables more possible ways of substitution.

The computational time of the DP algorithm is reported in Table 2, from which we can see that the DP is very efficient when m is small. As m and T increase, the memory space requirement for storing the values of function F_t and the optimal policy grows substantially and becomes a limitation of the DP algorithm.

The above data setting also enables us to perform a sensitivity analysis on the conversion cost factor (μ), the holding cost factor (λ), the variability of demand (α_{time}, α_{prdt}), and the variability of cost parameters (β_{time}, β_{prdt}). In this part of the experiments, we restrict the experiments to the case of $m = 2$ and $T = 10$. Note that from the first part, this restriction has little effect on the results for R^H, but that R^W deteriorates when m becomes large. In order to investigate the effect of the conversion cost factor μ, for each value of μ we let $\lambda = 0.25, 0.5, 1$ and $\alpha_{\text{time}}, \alpha_{\text{prdt}}, \beta_{\text{time}}, \beta_{\text{prdt}} = 0, 0.1, 0.2$. Thus, there are 243 sets of values of $(\lambda, \alpha_{\text{time}}, \beta_{\text{time}}, \alpha_{\text{prdt}}, \beta_{\text{prdt}})$ for each μ. For each set of $(\lambda, \alpha_{\text{time}}, \beta_{\text{time}}, \alpha_{\text{prdt}}, \beta_{\text{prdt}})$, we generate the production cost and demand data using the method in the first part, which is repeated ten times. Thus, for each μ, a total of $243 \times 10 = 2430$ instances are computed, and the average values of R^H and R^W are summarized in Table 3. Similarly, the effects of the holding cost factor and the variability of the cost and demand parameters are summarized in Tables 4–8.

From Tables 3–8, we have the following observations: (1) As the conversion cost factor increases (that is, as conversion gets more expensive), R^W decreases significantly, while there is little effect on R^H. (2) As the holding cost factor decreases, R^W decreases significantly, while the effect on R^H is less significant. (3) R^H increases as the variability of either the cost or demand parameter increases, but those parameters have little impact on R^W. The first observation coincides with the intuition that small conversion costs lead to significant potential cost savings if either the optimal DP solution or Heuristic H is used, compared to the solution obtained by simply ignoring the possibility of substitution. The second observation can be explained by the fact that product substitution yields savings mainly on part of the holding cost and setup cost. As λ decreases, from our data setting, both the holding and setup costs become less important than the production cost. This indicates that the benefit of product substitution diminishes as the production cost becomes the dominant cost component. The third observation points out a limitation of the use of Heuristic H. Namely, when the data are highly volatile, Heuristic H may not be a good choice. This is consistent with the fact that the Silver–Meal heuristic is mainly designed for stationary cost factors and is expected to perform poorly with rapid cost changes (see p. 91 of Zipkin 2000). In conclusion, the numerical results suggest that our

DP algorithm performs efficiently when the number of products is small, while Heuristic H is a good choice when the data are less volatile. They also indicate that it can be quite costly to ignore the effect of substitution, particularly when the conversion costs are small.

5. Conclusions

We have studied two dynamic lot size problems with one-way product substitution. Both problems are NP-hard in general. However, when the number of products is fixed, the dynamic programming algorithms developed in Sections 2 and 3 solve the problems in polynomial time. We remark that in many real world applications, the number of products that are hierarchically substitutable is typically small compared to the length of the planning horizon. Our models can be efficiently solved in these situations. When the number of products is large and the cost parameters do not vary significantly over time, the extended Silver–Meal heuristic provides a good approximate solution.

Recall that in **SWCP**, there are variable conversion costs, but there is no fixed cost associated with the conversion of products. In fact, the DP algorithm presented in Section 2 can be modified to handle fixed conversion costs by extending the state space. Denote B_{tjk} as the fixed cost of converting product P_j to product P_k in period t. We denote $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}; \gamma_{t1}, \gamma_{t2}, \ldots, \gamma_{tm})$ (where $\pi_{tj} \leq \gamma_{tj} \leq j$) as a restricted version of **SWCP** that satisfies the same conditions (i)–(iii) as those of $SWCP_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm})$, with the following additional condition:

(iv) For $j = 1, 2, \ldots, m$, if $\gamma_{tj} < j$ then a conversion from $P_{\gamma_{tj}}$ to P_j takes place in period t, otherwise (i.e., $\gamma_{tj} = j$) the demand of P_j in period t is satisfied by either the inventory carried from period $t-1$ or by the production that takes place in period t.

This restricted problem can be solved via a similar recursion as equation (1), while the total fixed conversion and production setup cost incurred in period t is given as

$$
\begin{aligned}
& s_t(\tau_{t1}, \tau_{t2}, \ldots, \tau_{tm}; \pi_{t1}, \pi_{t2}, \ldots, \pi_{tm}; \gamma_{t1}, \gamma_{t2}, \ldots, \gamma_{tm}) \\
& \quad = \sum_{\pi \in \{\pi_{tk} \mid \tau_{tk} = t;\, k=1,\ldots,m\}} K_{t\pi} + \sum_{k=1,\ldots,m \text{ s.t. } \gamma_{tk} < k} B_{t\gamma_{tk}k}.
\end{aligned}
$$

The computational complexity of this extended DP algorithm is $O\big((m+1)!(m!)^2 n^{m+1}\big)$. When the number of products m is fixed, the running time of this algorithm is the same as that presented in Section 2.

We conclude the paper by offering a few possible future research directions. First, our models can be extended to handle additional constraints that arise in real world applications, such as finite product life and limited number of conversions as discussed in Swaminathan and Kucukyavuz (2001), as well as capacity restrictions in production and conversion. Second, the cost functions of our models can be extended to general concave cost functions. However, our dynamic programs cannot be used to solve problems with such a general cost structure. The development of efficient algorithms for solving **SWCP** and **SWOP** with general concave costs is an interesting future research direction. Third, our models have assumed that all cost parameters are time dependent. One interesting question for future research is to investigate whether **SWCP** and **SWOP** can be solved more efficiently if some cost parameters are time independent. Fourth, one could study a multi-product DLS problem with one-way substitution for perishable products. This could be an extension of the model studied by Hsu (2000). Finally, it is desirable to study the multi-product DLS problem with substitution in both directions between products.

Appendix

Proof of Theorems 1 and 2: Recall that **SWOP** is a special case of **SWCP** and that the transformation from **SWOP** to **SWCP** can be done in polynomial time. Hence, it suffices to prove the strongly NP-hardness of **SWOP**. We transform the Exact Cover by 3-Sets

(X3C) problem to the decision version of **SWOP**. Given a set $A = \{a_1, a_2, \ldots, a_{3q}\}$ and a collection $C = \{A_1, A_2, \ldots, A_r\}$ of 3-element subsets of A, the X3C problem asks whether there exists a subcollection $C' \subseteq C$ such that every element of A occurs in exactly one member of C'. The X3C problem is known to be NP-hard in the strong sense (Garey and Johnson 1979).

Given an arbitrary instance of X3C, we construct a corresponding instance of the decision version of **SWOP** as follows: Let $m = r + 1$ and $T = 6q$. Denote $A_{r+1} = \emptyset$. Let

$$p_{tj} = 0 \quad (t = 1, 2, \ldots, T;\ j = 1, 2, \ldots, m);$$

$$D_{tj} = \begin{cases} 1, & \text{if } t \text{ is even and } j = m, \\ 0, & \text{otherwise,} \end{cases} \qquad (t = 1, 2, \ldots, T;\ j = 1, 2, \ldots, m);$$

$$K_{tj} = \begin{cases} 1, & \text{if } t = 1 \text{ and } 1 \le j \le m-1, \\ +\infty, & \text{otherwise,} \end{cases} \qquad (t = 1, 2, \ldots, T;\ j = 1, 2, \ldots, m);$$

$$h_{tj} = \begin{cases} 0, & \text{if } t \text{ is odd and } a_{(t+1)/2} \in A_j, \\ 2M, & \text{if } t \text{ is even and } a_{t/2} \in A_j, \\ M, & \text{otherwise,} \end{cases} \qquad (t = 1, 2, \ldots, T-1;\ j = 1, 2, \ldots, m);$$

threshold, $L = 3q(3q-1)M + q$;

where M is any integer greater than q. Note that the holding costs are defined in such a way that if one unit of product is held from period $2t-1$ until period $2t+1$, then the cost of holding must be $2M$, for $t = 1, 2, \ldots, 3q-1$. Obviously, the above construction can be done in polynomial time. We will show that there exists $C' \subseteq C$ such that every element of A occurs in exactly one member of C' if and only if there exists a solution to **SWOP** with a total cost of no more than the threshold value L.

Suppose there exists $C' \subseteq C$ such that every element of A occurs in exactly one member of C'. For each $A_j \in C'$, let $A_j = \{a_{\ell_1}, a_{\ell_2}, a_{\ell_3}\}$, where $\ell_1 < \ell_2 < \ell_3$. Then for $t = 1, 2, \ldots, T$ and $j, k = 1, 2, \ldots, m$, we set

$$x_{tj} = \begin{cases} 3, & \text{if } t = 1 \text{ and } A_j \in C', \\ 0, & \text{otherwise;} \end{cases}$$

$$I_{tj} = \begin{cases} 3, & \text{if } 1 \le t \le 2\ell_1 - 1 \text{ and } A_j \in C', \\ 2, & \text{if } 2\ell_1 \le t \le 2\ell_2 - 1 \text{ and } A_j \in C', \\ 1, & \text{if } 2\ell_2 \le t \le 2\ell_3 - 1 \text{ and } A_j \in C', \\ 0, & \text{otherwise;} \end{cases}$$

$$y_{tjk} = \begin{cases} 1, & \text{if } (t = 2\ell_1 \text{ or } t = 2\ell_2 \text{ or } t = 2\ell_3) \text{ and } A_j \in C' \text{ and } k = m, \\ 0, & \text{otherwise.} \end{cases}$$

Note that in the above instance of **SWOP**, positive demands only appear in product P_m and in time periods $2, 4, \ldots, 6q$. Since every element of A occurs in exactly one member of C', for each positive demand $D_{2k,m}$ ($k = 1, 2, \ldots, 3q$), there exists $A_j \in C'$ such that $a_k \in A_j$. Thus, in the constructed solution, demand $D_{2k,m}$ will be satisfied by product P_j, which is produced in period 1, held until period $2k$, and used to substitute P_m in period $2k$. We note that in the constructed solution, the holding cost of satisfying the demand $D_{2k,m}$ is equal to $2(k-1)M$, for $k = 1, 2, \ldots, 3q$. Hence, the total holding cost is equal to $\sum_{k=1}^{3q} 2(k-1)M = 3q(3q-1)M$. The total setup cost is equal to $|C'| = q$, and the total production cost is zero. Thus, the total cost of this constructed solution is $3q(3q-1)M + q$, which is exactly L.

Conversely, if there exists a solution to **SWOP** with a total cost no more than L, then in this solution the production can only take place in period 1 to produce $P_1, P_2, \ldots, P_{m-1}$. Let

$$J = \{j \mid x_{1j} > 0;\ j = 1, 2, \ldots, m-1\}.$$

Thus, J is the index set of j where the production of P_j takes place in period 1. We define

$$C' = \{A_j \mid j \in J\}.$$

Note that for $k = 1, 2, \ldots, 3q$, the holding cost of carrying one unit of any product from period 1 through period $2k-1$ is $2(k-1)M$. Hence, in any feasible solution with a total cost of no more than L, the holding cost needed to satisfy demand $D_{2k,m}$ is at least $2(k-1)M$. Therefore, the total holding cost in any feasible solution is at least $\sum_{k=1}^{3q} 2(k-1)M = 3q(3q-1)M$. This implies that the total setup cost in that solution is no more than q, or equivalently, $|C'| \le q$. Meanwhile, for every $a_k \in A$ ($k = 1, 2, \ldots, 3q$), there exists $A_j \in C'$ such that $a_k \in A_j$. This is because if a_k did not belong to any member of C', then in the solution to **SWOP**, the holding cost needed to satisfy demand

$D_{2k,m}$ would be at least $2(k-1)M + M$, and the total holding cost would be at least $3q(3q-1)M + M > L$, which is a contradiction.

This completes the proof of Theorems 1 and 2. ∎

Acknowledgment

The authors would like to thank four anonymous referees for their helpful comments and suggestions.

References

Aggarwal, A. and Park, J.K. (1993) Improved algorithms for economic lot size problems. *Operations Research,* **41,** 549–571.

Ahuja, R.K., Magnanti, T.L. and Orlin, J.B. (1993) *Network Flows: Theory, Algorithms, and Applications,* Prentice Hall, Englewood Cliffs, NJ.

Bassok, Y., Anupindi, R. and Akella, R. (1999) Single period multiproduct inventory models with substitution. *Operations Research,* **47,** 632–642.

Chan, L.M.A., Muriel, A., Shen, Z.-J. and Simchi-Levi, D. (2002) On the effectiveness of zero-inventory-ordering policies for the economic lot-sizing model with a class of piecewise linear cost structures. *Operations Research,* **50,** 1058–1067.

Chand, S., Ward, J.E. and Weng, Z.K. (1994) A parts selection model with one-way substitution. *European Journal of Operational Research,* **73,** 65–69.

Drezner, Z., Gurnani, H. and Pasternack, B.A. (1995) An EOQ model with substitutions between products. *Journal of the Operational Research Society,* **46,** 887–891.

Erickson, R.E., Monma, C.L. and Veinott, A.F. (1987) Send-and-split method for minimum-concave-cost network flows. *Mathematics of Operations Research,* **12,** 634–664.

Garey M.R. and Johnson, D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness,* Freeman, New York, NY.

Guisewite G.M. and Pardalos, P.M. (1993) A polynomial time solvable concave network flow problem. *Networks,* **23,** 143–147.

Gurnani, H. and Drezner, Z. (2000) Deterministic hierarchical substitution inventory models. *Journal of the Operational Research Society,* **51,** 129–133.

Herer, Y.T. and Tzur, M. (2001) The dynamic transshipment problem. *Naval Research Logistics,* **48,** 386–408.

Hsu, A. and Bassok, Y. (1999) Random yield and random demand in a production system with downward substitution. *Operations Research,* **47,** 277–290.

Hsu, V.N. (2000) Dynamic economic lot size model with perishable inventory. *Management Science,* **46,** 1159–1169.

Jones, P.C., Lowe, T.J., Muller, G., Xu, N., Ye, Y. and Zydiak, J.L. (1995) Specially structured uncapacitated facility location problems. *Operations Research,* **43,** 661-669.

Lamar, B.W. (1993) An improved branch and bound algorithm for minimum concave cost network flow problems. *Journal of Global Optimization,* **3,** 261–287.

Lee, S.-B. and Luss, H. (1987) Multifacility-type capacity expansion planning: algorithms and complexities. *Operations Research,* **35,** 249–253.

Pentico, D.W. (1974) The assortment problem with probabilistic demands. *Management Science,* **21,** 286–290.

Pentico, D.W. (1976) The assortment problem with nonlinear cost functions. *Operations Research,* **24,** 1129–1142.

Rao, U.S., Swaminathan, J.M. and Zhang, J. (2004) Multi-product inventory planning with downward substitution, stochastic demand and setup costs. *IIE Transactions,* **36,** 59–71.

Silver, E.A. and Meal, H.C. (1973) A heuristic for selecting lot size quantities for the case of a deterministic time-varying demand rate and discrete opportunities for replenishment. *Production and Inventory Management,* **14,** 64–74.

Smith, S.A. and Agrawal, N. (2000) Management of multi-item retail inventory systems with demand substitution. *Operations Research,* **48,** 50–64.

Swaminathan, J.M. and Kucukyavuz, S. (2001) Utilizing postponement and downward substitution for managing perishable inventory. Working paper.

Tuy, H., Ghannadan, S., Migdalas, A. and Varbrand, P. (1995) The minimum concave cost network flow problem with fixed numbers of sources and nonlinear arc costs. *Journal of Global Optimization,* **6,** 135–151.

Veinott, A.F. (1969) Minimum concave-cost solution of Leontief substitution models of multi-facility inventory systems. *Operations Research,* **17,** 262–291.

Wagner, H.M. and Whitin, T.M. (1958) Dynamic version of the economic lot size model. *Management Science,* **5,** 89–96.

Ward, J.A. (1999) Minimum-aggregate-concave-cost multicommodity flows in strong-series-parallel networks. *Mathematics of Operations Research,* **24,** 106–129.

Zangwill, W.I. (1968) Minimum concave cost flows in certain networks. *Management Science,* **14,** 429–450.

Zipkin, P.H. (2000) *Foundations of Inventory Management,* McGraw-Hill, Boston.

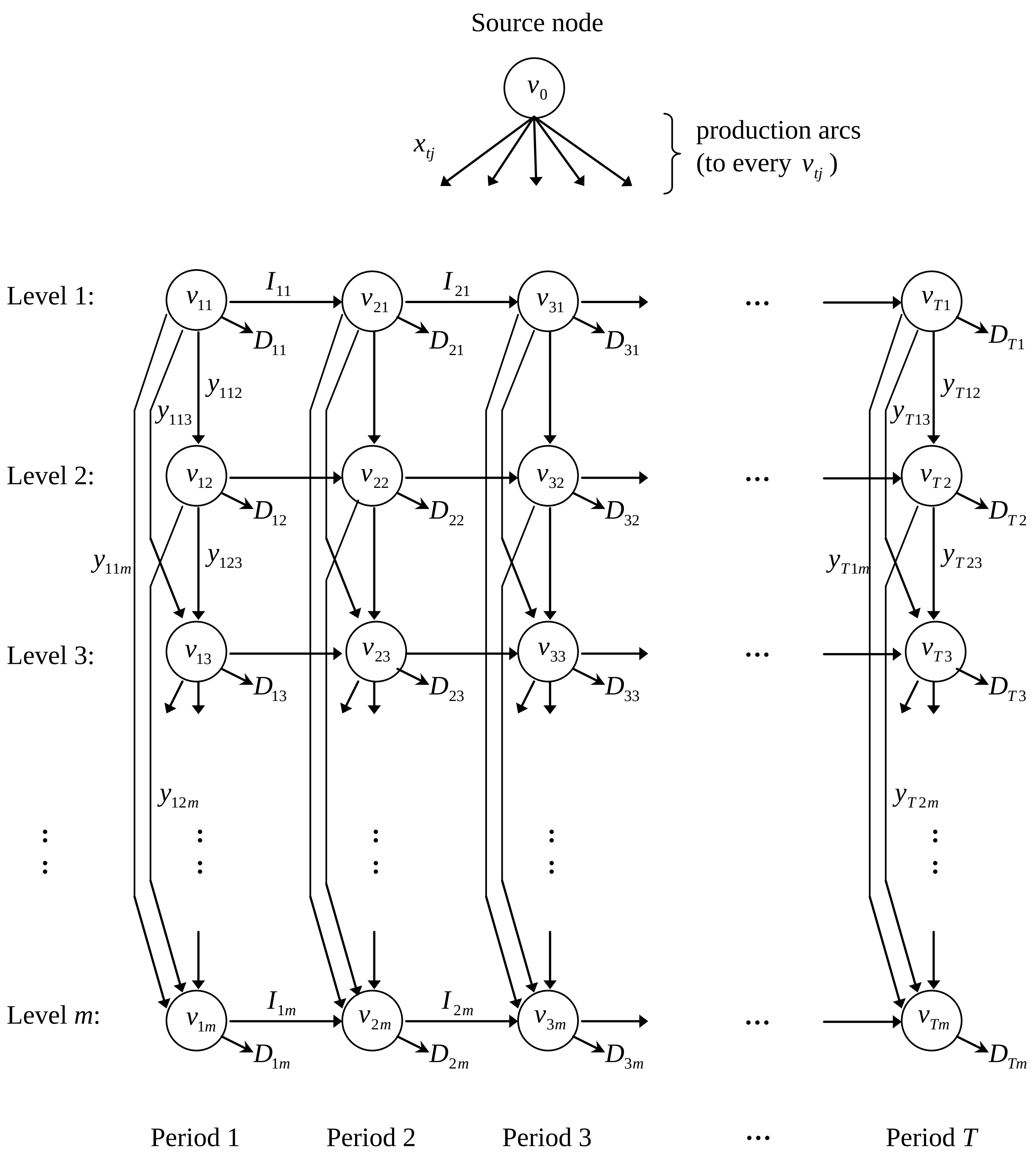

Figure 1. Network G for problem **SWCP**.

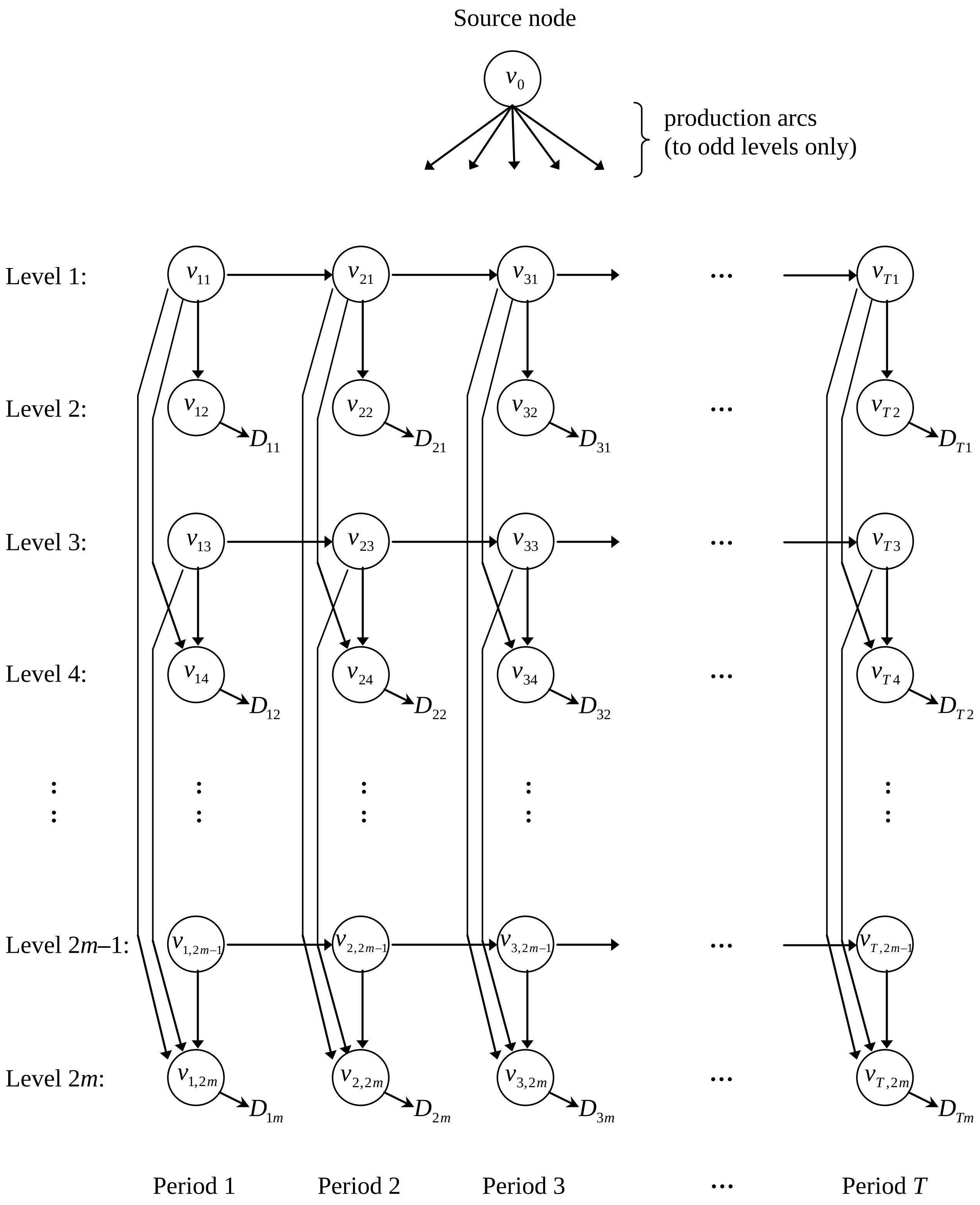
Source node
v_0
production arcs
(to odd levels only)
Level 1:
Level 2:
Level 3:
Level 4:
Level 2m–1:
Level 2m:
v_{11} v_{21} v_{31} v_{T1}
v_{12} v_{22} v_{32} v_{T2}
v_{13} v_{23} v_{33} v_{T3}
v_{14} v_{24} v_{34} v_{T4}
$v_{1,2m-1}$ $v_{2,2m-1}$ $v_{3,2m-1}$ $v_{T,2m-1}$
$v_{1,2m}$ $v_{2,2m}$ $v_{3,2m}$ $v_{T,2m}$
D_{11} D_{21} D_{31} D_{T1}
D_{12} D_{22} D_{32} D_{T2}
D_{1m} D_{2m} D_{3m} D_{Tm}
Period 1
Period 2
Period 3
…
Period T

Figure 2. Network diagram showing that **SWOP** is a special case of **SWCP**.

Table 1. Stability results

	$m=2$ $T=10$	$m=2$ $T=20$	$m=2$ $T=40$	$m=2$ $T=80$	$m=3$ $T=10$	$m=3$ $T=20$	$m=3$ $T=40$
R^H	7.4%	8.1%	8.4%	8.7%	9.8%	10.4%	9.6%
R^W	11.9%	11.8%	11.7%	11.5%	20.1%	19.9%	19.5%

Table 2. CPU time (in sec.) of the DP algorithm for SWCP

	$m=2$ $T=10$	$m=2$ $T=20$	$m=2$ $T=40$	$m=2$ $T=80$	$m=3$ $T=10$	$m=3$ $T=20$	$m=3$ $T=40$
DP for SWCP	0.001	0.016	0.031	0.203	0.016	0.25	3.93

Table 3. Sensitivity analysis of the conversion rate

	$\mu=0$	$\mu=0.01$	$\mu=0.02$
R^H	7.4%	7.2%	7.1%
R^W	17.3%	14.0%	10.8%

Table 4. Sensitivity analysis of the holding cost rate

	$\lambda=0.25$	$\lambda=0.5$	$\lambda=1$
R^H	8.1%	7.2%	6.5%
R^W	5.5%	13.0%	23.6%

Table 5. Sensitivity analysis of the demand variability over time

	$\alpha_{\text{time}}=0$	$\alpha_{\text{time}}=0.1$	$\alpha_{\text{time}}=0.2$
R^H	6.2%	7.4%	8.5%
R^W	13.8%	14.1%	14.2%

Table 6. Sensitivity analysis of the production cost variability over time

	$\beta_{\text{time}}=0$	$\beta_{\text{time}}=0.1$	$\beta_{\text{time}}=0.2$
R^H	4.4%	6.5%	11.0%
R^W	13.8%	14.3%	13.8%

Table 7. Sensitivity analysis of the demand variability across products

	$\alpha_{\text{prdt}} = 0$	$\alpha_{\text{prdt}} = 0.1$	$\alpha_{\text{prdt}} = 0.2$
R^H	6.8%	7.2%	8.1%
R^W	13.9%	14.1%	14.1%

Table 8. Sensitivity analysis of the production cost variability across products

	$\beta_{\text{prdt}} = 0$	$\beta_{\text{prdt}} = 0.1$	$\beta_{\text{prdt}} = 0.2$
R^H	6.2%	6.8%	9.0%
R^W	14.4%	13.9%	13.6%