Placing Safety Stock in Logistic Networks under Guaranteed-Service Time Inventory Models: An Application to the Automotive Industry

L. A. Moncayo-Martínez*[1,2], E. O. Reséndiz-Flores[2], D. Mercado[2] and C. Sánchez-Ramírez[3]

[1] Department of Industrial and Operation Engineering
Instituto Tecnológico Autónomo de México (ITAM)
México, D. F., México
*luis.moncayo@itam.mx
[2] Division of Postgraduate Studies and Research
The Technological Institute of Saltillo
Saltillo, Coahuila, México
[3] División de Ciencias e Ingeniería
Department of Industrial Engineering & Manufacturing Science
Orizaba Institute of Technology
Orizaba, Veracruz, México

ABSTRACT
The aim of this paper is to solve the problem of placing safety stock over a Logistic Network (LN) that is represented by a Generic Bill of Materials (GBOM). Thus the LN encompasses supplying, assembling, and delivering stages. We describe, in detail, the recursive algorithm based on Dynamic Programming (DP) to solve the placing safety stock problem under guaranteed-service time models. We also develop a java-based application (JbA) that both models the LN and runs the recursive DP algorithm. We solved a real case of a company that manufactures fixed brake and clutch pedal modules of cars' brake system. After running JbA, the levels of inventory decreased by zero in 55 out of 65 stages.

Keywords: Safety Stock, Guaranteed-service time, Dynamic Programming, Automotive Industry.

RESUMEN
El objetivo de este artículo es resolver el problema de colocación inventario en una Red Logística (LN) que es representada por una Lista de Materiales Genérica (GBOM), de manera que la LN tiene etapas de suministro, ensamble y entrega. Describimos, a detalle, el algoritmo recursivo de Programación Dinámica (DP) para resolver el problema de colocación de inventario en modelos de servicio garantizado. También programamos una aplicación en java (JbA) que modela la LN y ejecuta las operaciones recursivas del algoritmo de DP. Resolvimos un caso real de una empresa que manufactura módulos de frenos y pedales del clutch del sistema de frenos utilizados en los autos. Los resultados muestran que los niveles de inventarios se reducen a cero en 55 de 65 etapas después de ejecutar nuestra JbA.

1. Introduction

Manufacturing companies are highly pressured into producing quality products and delivering them to the right location, at the right quantity or amount, and at the right place, subject to reduce both manufacturing and logistic costs. In order to reach this aim, companies have realised that a global approach is required to coordinate operations across the entire Logistic Network (LN) or Supply Chain, e.g. share information to minimise the bullwhip effect [1]; pass products' demand to upstream members to reduce inventory levels [2] or solve the routing and inventory problem simultaneously [23,26]. Moreover, companies have to dynamically evaluate the LN operations [24] and reduce the complexity generated by the product diversification [25] to reach the global aim of cost reduction.

Global inventory management is an important strategy in reducing manufacturing and logistic costs because a proper inventory policy could result in reducing the amount of safety and pipeline stock.

In literature, the problem of placing inventory is divided into single stage and multi stages. The first one is a difficult but well studied problem, the models used to solve it are deterministic (e.g. economic order quantity and wagner-whitin model) and stochastic (e.g. (r,Q) and (s,S) policies)[3]. The multi stage problem could be either stochastic-service (SS) or guaranteed-service (GS). The main difference between SS and GS is the way in which a stage supplies components or assemblies to other downstream stages.

Backorders are allowed in SS multi stage problem, i.e. a fraction of an order cannot be filled at the right time due to a lack of available supply [4-5]. Unlike SS model, the GS model must serve the complete order just in a guaranteed-service time ω.

Our paper deals with GS models in multi stages, thus the problem is to minimise the cost of the safety stock that every stage must hold in order to serve its downstream stages just in the ω, given that the days of inventory required are $U = \delta + t - \omega$, where δ is the time in which a stage must be served by its upstream stages and t is the time spent by a stage to perform its task.

The novelties of the proposed paper lie in the methodology employed to solve a real-life LN and in the java-based application programmed to solve the DP algorithm used to solve the GS inventory placing problem [2]. Additionally, we provide a pseudo code full of practical insights to carry out the recursive operations.

We implemented and applied the GS time inventory model (GSTIM) to a company that manufactures fixed brake and clutch pedal modules. We both selected the product with the highest demand and described the steps followed to collect the necessary information to run the java-based application.

In the following section, a literature review of the GSTIM is provided. In section 3, the model is defined and some assumptions are stated. In section 4, the methodology used to implement the GS model is depicted, so also the DP algorithm and the java-based application are described. A real case is described in section 5. Finally, results are presented in section 6 and we draw some conclusions in section 7.

2. Related Literature

In this section, we cite a set of approaches related to GSTIM. Back in 1958, Simpson [6] solved the problem of placing inventory over a serial process. Adjacent stages were coupled together to equate the incoming service time of a downstream stage with the outbound service time of its upstream stage. The optimum inventory level per stage was found by determining the service time. It was proven that the optimal service time in serial processes is found in an extreme point property where the outgoing service time is equal to either zero or its incoming service time plus its processing time, i.e. using an all-or-nothing inventory policy. A boundary demand is used, thus it is interpreted as the amount of inventory a company wants to satisfy from its safety stock.

Later, the same problem was solved by standard operations of DP in [7] and was extended to supply chains modelled as assembly networks [8], to distribution networks [9], and to spanning trees [10].

In a recent approach, a stage could include more than one upstream or downstream stage [2,11], so we have to notice two important facts: i) in case a downstream stage is served by multiple upstream stages, the downstream stage has to wait for the component with the longest service time, and ii) in case an upstream stage serves multiple downstream stages, the upstream stage quotes the same service time to all the adjacent downstream stages. Moreover, the assumption about demand boundary remains and it is supposed that the LN is designed already, thus the time and cost of every stage is known.

The complexity of the aforementioned approach has been proven to be NP-hard [12, 13]. As a result, modification to the DP algorithms have appeared in literature to solve bigger instances than those solved efficiently using the DP standard algorithm, e.g. CPLEX is used to iteratively solve a piecewise-linear demand once redundant constrains are added [14]; branch and bound algorithm is used to reduce complexity [15]; tailor-made heuristic has been proposed [16]; and general purpose genetic algorithms are used to solve the problem [17]. Other generalizations that do not apply to our real-life case included: capacity constraints [18], LN design constraints [19], non-stationary demand [20], and stochastic lead times [21].