Int j simul model 11 (2012) 1, 40-57
ISSN 1726-4529 Original scientific paper

A HIERARCHICAL PRODUCTION PLANNING SYSTEM SIMULATOR

White, L. R.
Eastern Illinois University, Lumpkin College of Business and Applied Sciences,
600 Lincoln Avenue, Charleston, IL 61920, United States
E-Mail: lrwhite2@eiu.edu

Abstract
This research develops and demonstrates a Hierarchical Production Planning System Simulator (HPPSS) for the analysis of feedback-and-control mechanisms between linear programming models at two levels within a Hierarchical Production Planning (HPP) system. This work focuses on planning levels dealing with tactical decisions. Three areas distinguish it from prior research. First, the HPPSS allows explicit examination of the effects of different information exchanges between the levels of a hierarchical model. Second, the HPPSS allows examination of the problems for which a given feedback-and-control mechanism performs well. Finally, the HPPSS allows the effects of a rolling horizon implementation on the hierarchical models of the production planning problem to be investigated. The significance is that a more thorough understanding of the costs and benefits of various mechanisms for information exchange between the levels of hierarchical models of planning problems over time will lead to improved hierarchical decision-making techniques that may influence organizational design.
(Received in May 2011, accepted in October 2011. This paper was with the author 1 month for 1 revision.)

Key Words: Distributed Decision Making, Hierarchical Production Planning, Hierarchical Modelling, Simulation, Feedback and Control, Information Exchange

1. INTRODUCTION

The complexity of managing a business with its multitude of decisions that must be made over multiple planning horizons and varying levels of detail has led naturally to today's hierarchically structured organizations. Similarly, efforts to model the organization's decision-making have evolved into hierarchies of models. Since the groundbreaking work on hierarchical planning and scheduling in a production environment [1], numerous researchers and practitioners have developed hierarchical models for production planning and other fields. The hierarchical modelling concept has proven to be a fruitful area of research and application to this day, with numerous models being compiled and summarized in recent books [2-4]. Reviews of the field identify feedback as one of the primary areas of hierarchical models that merits further research [5-6]. It has been suggested that simulation studies can be used to help select an appropriate set of models including production planning in particular situations [7]. It has also been observed that the hierarchical planning approach requires "further basic research and computational experimentation" [8]. Nevertheless, the analysis of feedback-and-control mechanisms within hierarchical models has largely been ignored. A notable exception is the use of simulation to evaluate the impact of using feedback to dynamically update lead time estimates on performance of a two-level hierarchical planning system and identify conditions under which frequent updates are detrimental [9].

The hierarchical models used in this research will differ from those described in prior research in two significant areas: (1) the temporal decomposition and (2) the feedback-and-control mechanisms governing interrelationships between levels. However, the primary

contribution of this research is not the development of a unique temporal decomposition or new feedback and control mechanisms although many of the concepts described here are not found in the hierarchical planning literature. The concepts described here are pragmatic ideas that are almost unavoidably required in practice. Rather, the primary contribution of this research is the development of the HPPSS that can be used to compare the various feedback and control mechanisms in a realistic temporal decomposition. Discussion of the HPPSS is reserved for Sections 3 and 4 while the concepts incorporated in it are discussed in more detail in Section 2 below. An example is given in Section 5 with conclusions in Section 6.

2. CONCEPTUAL DEVELOPMENT

2.1 Temporal decomposition

The temporal decomposition used in this research differs from that used in most prior research in three respects. These distinctions are described here and illustrated in Fig. 1. First, most prior research uses the same planning periods at each level of the decision-making hierarchy [e.g., 10-15]. This is illustrated in Fig. 1a where the lower and upper levels use the same planning period. Standard decomposition techniques of mathematical programming such as Dantzig-Wolfe decomposition or Benders decomposition [16] would fall into this category as they employ the same variables at all levels of decomposition. In contrast, the current research is more closely aligned with the concepts described in other research in recognizing that a primary motivation for decomposition in production planning and scheduling is the different time scales that are relevant to different decisions [17]. These different time scales naturally lead to the use of a hierarchy with each decision-making level using a planning period relevant to the particular decisions under its control. Higher decision-making levels generally use longer planning periods than lower decision-making levels. This is illustrated in Fig. 1b where the lower level planning periods are shorter than the upper level planning periods. It should be observed from Fig. 1 that both the typical temporal decompositions from the

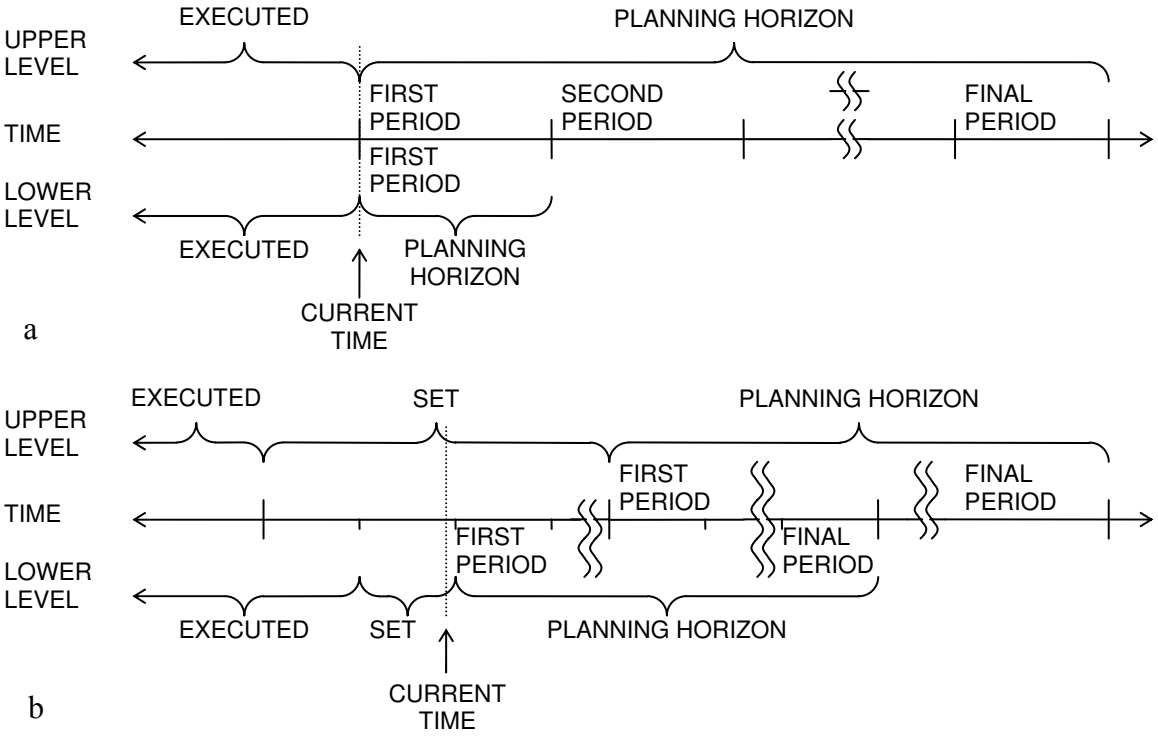

Figure 1: a) Typical temporal decomposition, b) Temporal decomposition for HPPSS.

existing literature and the decompositions used in this research generally use a longer planning horizon for the upper level than the lower level.

The second distinction from prior temporal decompositions is related to the first but goes beyond merely recognizing the different time scales [17]. In prior research in HPP, the planning horizon used at a lower level is a subset of the planning horizon used at the next higher level. In fact, in most prior research the lower-level planning horizon is simply taken as the first planning period from the next higher level as shown in Fig. 1a. Similarly, decomposition methods of mathematical programming require the lower-level planning horizon to be a subset of the upper-level planning horizon as they rely on the structure of the overall mathematical model to break the "lower level" into a number of subproblems spanning the entire horizon considered at the "upper level." However, in the current research we recognize that the use of shorter planning periods on a lower level will generally cause the lower level's planning horizon to include some time prior to the start of the next higher level's planning horizon as shown in Fig. 1b. These overlapping planning horizons are a direct result of the longer planning periods used at higher levels and the corresponding need to lock in decisions at those higher levels further in advance than decisions at the lower levels. Related to this, we recognize that the end of the lower level's planning horizon would typically correspond to the end of a planning period at the next higher level for coordination purposes although such correspondence may not be necessary from a modelling standpoint.

The third distinction is that most prior research assumes that the state of the system is known with certainty at the beginning of the planning horizon. In many instances, this requires instantaneous reporting and decision making at the conclusion of the period preceding the current planning horizon. This is illustrated in Fig. 1a where the current time is on the boundary between the past, which has been shaped by previously executed decisions, and the planning horizon for which we are developing plans. The current research recognizes that plans must be made in a dynamic environment where the state of the system is constantly changing, and that reporting and decision making take time. Therefore, the current research uses an estimate of what the state of the system will be at the beginning of the planning horizon. This distinction, which is most significant in stochastic environments, is illustrated in Fig. 1b where the current time is placed within a period in which plans are considered set and are being executed. Even in deterministic problems such as those used in this research the state of the system at the start of the upper-level planning horizon cannot be known with certainty until all lower-level decisions preceding that time have been finalized. This distinction is closely related to the feedback/control distinctions discussed below.

2.2 Feedback/control mechanism

In this research, we distinguish between plans, decisions, and targets. These are all obtained from the optimal solution to a given level's model. We define plans to be the values of all the variables at a given level. Decisions are the values of those variables for which a given level has final decision-making responsibility and which impose hard constraints on lower levels. Targets are values based on a given level's plans and are passed to a lower level which is not forced to meet them but which may be given incentives to do so. The term "plans," as used here, encompasses both decisions and targets that are passed between levels. However, many plans need not be directly communicated between levels.

As shown in Fig. 2a, many of the hierarchical production planning models in the literature rely primarily, if not exclusively, on a single top-to-bottom pass with decisions at each level being passed to the next lower level as hard constraints [1, 10-12, 14]. This results in a rather rigid control structure. When these single-pass models employ feedback, it is generally used to modify an existing higher-level solution to permit a feasible solution to a lower-level model.

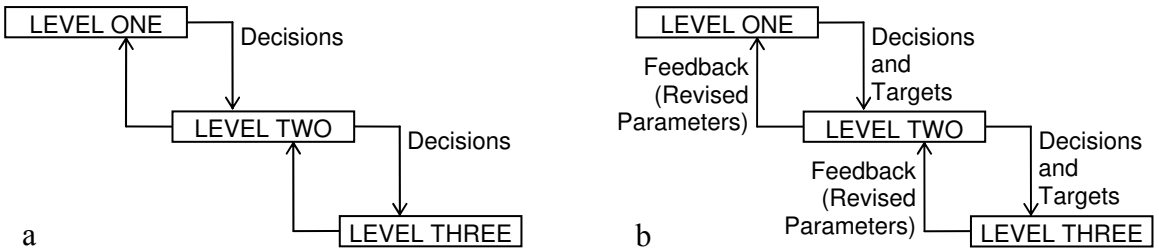

Figure 2: a) Typical top-down hierarchy, b) Hierarchy with feedback.

Early researchers recognized that passing production quantities as hard constraints from higher levels to lower levels could easily force shortages of specific items if the inventory of other items within the same product type (or family) was large enough to prevent a sufficient amount of the product type (or family) from being produced in the first period [10]. To eliminate these infeasibilities at the lower levels, they introduced the concept of "effective" demand whereby initial item inventories are depleted by satisfying actual demands prior to rolling up the aggregate demands. This use of effective demand adds another computational step in the planning process and requires detailed forecasts for a sufficiently long horizon to deplete all inventories.

Standard mathematical decomposition approaches such as Dantzig-Wolfe decomposition and Benders decomposition [16] employ an iterative mechanism with endogenously generated parameters as the coordination mechanism. Similarly, some production planning models have been solved by mathematical decomposition employing an iterative mechanism with endogenously generated parameters such as dual prices or Lagrange multipliers [e.g., 13] as the coordination mechanism. These models iterate between adjacent levels until convergence criteria are met. While endogenously generated parameters generally provide the marginal values (in some sense) of the resources, they do not necessarily reflect the costs and benefits to the decision maker as these are typically exogenous parameters. While this approach to production planning has sometimes been referred to as HPP, the definitions used in this research classify it as a mathematical decomposition of a production planning problem rather than as HPP. Unlike HPP, it uses the same variables at all levels and it relies on the structure of the overall problem to break the "lower level" into a number of subproblems spanning the entire horizon considered at the "upper level." Hence, it shares the data requirements of a monolithic model in requiring detailed information over the entire planning horizon.

In this research, we assume that a lower level has more detailed information (over a shorter planning horizon) than an upper level and an objective function in agreement with that of the upper level. One goal of this research is to examine the idea that the lower level should be given as much flexibility as possible in using the resources available to it to optimize the objective. Therefore, in addition to the rigid control structure used in prior research, the HPPSS developed in this research allows analysis of a very loose control structure. In this loose control structure, soft targets for the state of the system at the end of the lower-level's planning horizon are passed down from the upper level along with decisions directly under the control of the upper level. This passing of information from the upper to the lower level is shown in Fig. 2b. The use of targets for the end of the lower level's planning horizon is to prevent the lower level from becoming short sighted and to recognize the long-term view used by the upper level. In addition, the use of soft targets eliminates the need to use "effective" demand to prevent infeasibilities at the lower levels.

This research also recognizes that lower-level decisions frequently influence the exogenous parameters used by the upper-level model, especially when the upper-level model uses aggregate data as is common in hierarchical models. Therefore, feedback from the lower level is necessary to adjust the parameters used in the upper level's decision-making process.

In the case of aggregated data, this adjustment involves using the lower-level's tentative plans to adjust the weighting used to aggregate exogenous parameters. This feedback of revised parameters from the lower level to the upper level is shown in Fig. 2b.

Thus, as shown in Fig. 2b, except for the top and bottom levels each level within the hierarchy has two basic inputs and two basic outputs. The inputs are the information passed down from the higher level(s) and the feedback of revised parameters from the lower level(s). The outputs are the plans passed on to the lower level(s) and the feedback of revised parameters to the upper level(s). A more detailed view of the components required at a given level is shown in Fig. 3. We refer to this given level as Level L and identify three components that must be present for Level L to function properly. The core component is the Planning component. The inputs shown on Fig. 2b are routed directly to Level L Planning as shown in Fig. 3. Additionally, Fig. 3 shows exogenous parameters that are not under the control of any other level as inputs to Level L Planning. The final inputs to Level L Planning are previous Level L decisions that have been retained within Level L. Level L Planning has two outputs as shown in Fig. 3. First, the subset of Level L plans comprised of decisions and targets is sent to an implementation component for processing and forwarding to lower level(s). Second, the Level L plans are sent to a component that refines data dependent on Level L decisions for feedback to higher level(s).

The second Level L component is responsible for implementing Level L decisions. This component processes decisions and targets from higher levels, exogenous parameters, and Level L decisions and targets produced by Level L Planning to prepare the exogenous parameters, decisions, and targets to be passed down to the lower levels. The third Level L component is responsible for refining data dependent on plans made by Level L or lower and providing feedback to higher levels. Fig. 3 shows how these components are responsible for appropriate communications with other levels.

3. DESCRIPTION OF THE HPPSS

3.1 Overview of the Hierarchical Production Planning System Simulator

The purpose of the HPPSS is to quickly generate and solve numerous scenarios using monolithic, myopic, and hierarchical approaches so that the approaches may be compared. The HPPSS allows the study of combinations of models with feedback-and-control mechanisms. Therefore, unlike analytical proofs that must be done for every model, the HPPSS will allow us to derive principles which are applicable across a range of hierarchical production planning models. This contributes to the existing body of research in hierarchical production planning models by providing capabilities heretofore undeveloped for comparison

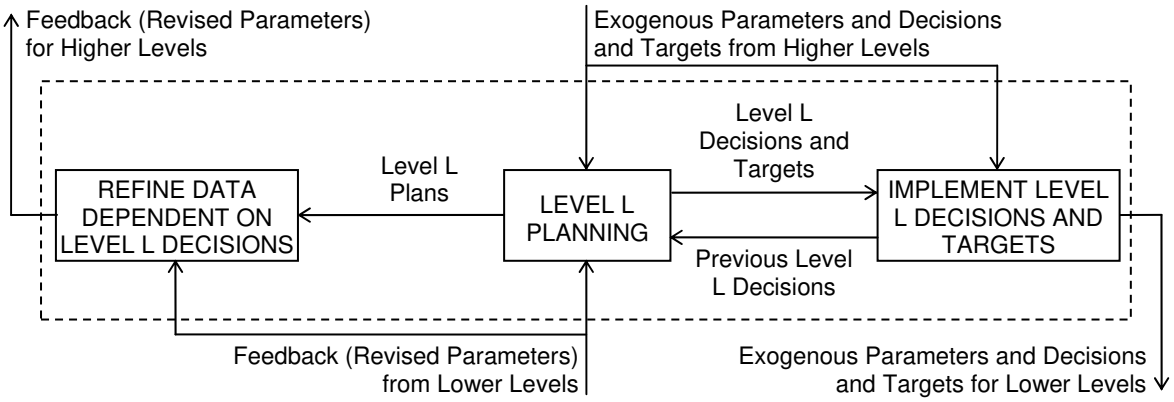

Figure 3: Feedback and control mechanism within a given level.

of various hierarchical approaches. Largely due to the difficulty in creating and programming even a single set of hierarchical models, most prior research has simply analysed one hierarchy for a particular problem without providing comparisons with alternative hierarchies. Similarly, few previous articles have discussed the losses to the organization resulting from the suboptimal nature of decisions made by hierarchical models. The simulator developed herein is a first step at providing a solid basis for comparison and analysis of hierarchical modelling approaches. While it does not address every conceivable hierarchical model or every possible type of feedback and control, it allows a wide variety of two-level hierarchical models for a range of variations on traditional production planning problems.

The HPPSS is a Fortran computer program containing the logic for transforming a set of input data into multiple, pseudo-randomly-generated, production planning scenarios and determining the resulting plans and profits for a specified variety of myopic and hierarchical models. A monolithic model is also created and solved optimally (if feasible) for each scenario. While it is not generally realistic to formulate, parameterize and solve a monolithic model in practice, in a simulation study a monolithic model provides a baseline against which myopic and hierarchical models can be compared since it provides the maximum profit that can be achieved under any planning system. The HPPSS allows a variety of different planning horizons to be used in the myopic and hierarchical approaches. It also allows several feedback and control mechanisms to be used in the hierarchical approach as will be discussed further below.

The HPPSS uses periods of three different durations. The shortest period is the lower-level planning period, henceforth also referred to (arbitrarily) as a week. As the name implies, the lower-level planning period is the planning period used in the lower level of a hierarchical model. It is also the planning period used by the monolithic and myopic models. The next shortest period is the upper-level planning period, henceforth also referred to (again arbitrarily) as a month. As the name implies, the upper-level planning period is the planning period used in the upper level of a hierarchical model. While not directly used as a planning period for the other models, the month is still used as an identifier to specify each week within those models (e.g., week 2 of month 3). The longest period is the cycle, which we will also refer to (arbitrarily as before) as a year. While not used as a planning period for any model, the year can be used as necessary to specify the week (e.g., year 2, month 3, week 2). The primary use of the cycle is to specify the periodicity with which to repeat the exogenous problem data such as demand, sales price, production cost, etc. In other words, for a problem that extends over multiple cycles (i.e., years), each year is identical to the first year with the exception of boundary conditions such as the initial levels of inventory, backorders, and workforce. The secondary use of the cycle is as an optional means of specifying the length of the upper-level planning horizon in a hierarchical model.

The HPPSS can analyse each scenario either for a specific number of cycles (in a transient problem) or until steady state is reached. Steady-state problems run until the state at the end of a cycle matches the state at the beginning of that cycle. The state is defined by the boundary conditions of inventory, backorders, and workforce. For the monolithic model, steady-state results can be immediately obtained by constraining terminal conditions to be equal to initial conditions and leaving both as decision variables [18].

At the core of the HPPSS is a master model. This master model contains elements that are common in many production planning problems. It is formulated as a deterministic linear program (LP). A description of the master model is given in Section 3.2. Not all elements of the master model are used in any single planning model, but the monolithic model, myopic models, and subproblems of the hierarchical models are all subsets of the master model. It is the development and use of this master model that enables the simulator to provide unparalleled versatility in comparing and analysing a wide range of myopic, hierarchical, and

monolithic models. Section 3.3 describes the model variations that can be solved based on the master model.

All LPs required by the models under investigation are formulated in terms of the master model and are solved using a version of the XMP subroutines for linear programming [19]. The structure of the Fortran program is such that only one subroutine in the HPPSS interacts with the linear programming subroutines so a different set of linear programming subroutines could easily be used in place of XMP. Other subroutines in the HPPSS program are responsible for reading the input data, pseudo randomly generating the requested scenarios, managing the data (input, generated, and results), formulating the required LPs for the various models and options, calculating the cumulative profit for each model and scenario combination, and reporting the results. The subroutine structure of the simulator will be described more fully in Section 4. Using the master model concept and standard linear programming subroutines allows the emphasis in the HPPSS to be on the solution approaches (monolithic, myopic, and hierarchical) and the various options relevant to those approaches rather than on different optimization models and routines.

3.2 Master model

As mentioned above, the monolithic model, myopic models, and subproblems of the hierarchical models are all are subsets of a master model that is formulated as a deterministic LP. The concept of the master model cannot be overemphasized. The master model allows the user of the simulator to focus on the feedback and control mechanisms rather than the formulation and solution of multiple submodels.

An overview of the master model is provided in Table I. The mathematical formulation of the master model is too lengthy for this article and can be found elsewhere [20]. Terminal inventory and/or backorder shortages and/or overages may be prohibited for some or all items. Similarly, terminal workforce shortage and/or overage may be prohibited. Constraints on aggregate goals are used by the lower level of hierarchical models only.

The demand, sales price, labour requirement, regular-time capacity (fixed or per worker), production time, production cost, and production yield parameters must be included in the data input into the simulator as must be the number of product groups, product items for each group, cycles (or steady-state indicator), months per cycle, and weeks for each month. All other problem parameters depend on the options available to the planner for inventory, backorders, overtime, hiring, and firing. Optional parameters include inventory and backorder costs, initial and terminal inventory and backorders, terminal backorder and inventory overage and shortage costs, inventory carryover rate, backorder retention rate, workforce attrition rate,

Table I: Overview of master model.

Objective: • Maximize Profit **Decision Variables:** • Sales • Production • Inventory • Backorders • Regular time • Overtime • Hiring • Firing	**Constraints:** • Material balance • Demand satisfaction • Capacity • Overtime limit • Workforce balance • Initial and terminal workforce or steady-state workforce • Initial and terminal inventory or steady-state inventory • Initial and terminal backorders or steady-state backorders • Aggregate sales goal • Aggregate production goal • Aggregate terminal inventory goal • Aggregate terminal backorder goal

initial and terminal workforce, terminal workforce overage and shortage costs, new worker efficiency, hiring and firing costs, regular-time costs, overtime capacity and overtime cost. The parameters are input into the simulator as probability distributions to facilitate the generation of multiple random scenarios. Available probability distributions include uniform (continuous), normal, exponential (shifted), gamma (shifted), beta (shifted and scaled), triangular, empirical, and deterministic.

3.3 Model variations

Based on the above information, the appropriate monolithic model is generated and solved for each scenario. Model instructions input by the user using available options specify which models to run in addition to the monolithic model. These additional models will be run only for scenarios in which an optimal solution exists to the monolithic model. The additional models are classified as either myopic models or hierarchical models. The hierarchical models are further classified as either single-pass models or iterative models. These models will be described more fully below.

The myopic models look at a short-term planning horizon with all details known over that horizon. The only information that the user must provide is the length of the planning horizon using one of four available methods. The planning horizons defined by three of the methods all terminate at the end of a month (fixed number of months, minimum number of weeks or maximum number of weeks). These are easily comparable to hierarchical models in which the lower-level planning horizon must extend through the end of an upper-level planning period in order for boundary conditions to be satisfied. The fourth method with a planning horizon of a fixed number of weeks is most relevant for studying the effects of planning horizon length on solution quality for myopic models rather than for comparison with hierarchical models. The initial conditions for a given myopic model are derived from the results of the preceding myopic model in the rolling horizon framework.

Hierarchical models require the user to specify considerably more information than myopic models. From the available options, the user must select the type of hierarchical model to be used (single-pass or iterative), the aggregate goals to be passed down from the upper level (sales, production, terminal inventory, terminal backorders, or combinations, and/or terminal workforce), the means of penalizing deviations from those goals (aggregate objective function coefficients, reduced costs, or prevent deviations unless required for feasibility), the means of aggregation of lower-level units into upper-level units (equal contributions, weight based on demand, or weight based on short-term plans), the lengths of the upper- and lower-level planning horizons (fixed number of cycles or months, minimum number of months or weeks, maximum number of months or weeks, or fixed number of months or weeks), and the levels (upper, lower, or both) at which overtime, hiring, and firing are considered.

The upper level of a hierarchical system considers only product groups and upper-level planning periods. It overlooks a lot of detail. The solution to the upper level provides constraints and/or targets that are used by the lower level. The lower level considers individual product items and lower-level planning periods. If included in the model, sales, production, inventory, backorders, and regular time are considered at both the upper and lower levels. Single-pass hierarchical systems are analysed to show the performance of hierarchical methods currently in the literature. The single-pass hierarchical systems each perform a single, top-to-bottom pass through the planning levels at each lower-level planning period. Iterative hierarchical systems iterate between the upper and lower levels to update the aggregate parameters used by the upper level during the time in which the planning horizons for the upper and lower levels overlap.

Myopic models and/or hierarchical systems are solved on a rolling-horizon basis until either the end of the transient period is reached or steady state is reached, whichever is applicable. Cumulative profits are calculated for the entire period (or steady-state cycle) covered by the scenario. These cumulative profits are directly comparable to the profits achieved by the monolithic model.

4. HPPSS COMPUTER PROGRAM

The HPPSS is a Fortran program with subprograms, as shown in Fig. 4. The main program, HIER, coordinates reading input files, generating scenarios, and solving various models.

The HPPSS uses two input files. The data input file contains information on the number of months, weeks, product groups, items, options available to the planner, and parameters for probability distributions for the problem data. The run information file contains the initial random number seed, the range of scenarios to be analysed, a convergence tolerance to be allowed for iterative models, a limit on the number of cycles to try before abandoning a steady state problem, a limit on the number of iterations to allow between levels during any week, whether to continue if convergence between levels is not achieved within the iteration limit, the level of reporting to output, and instructions for the models to be solved.

Internally, the HPPSS stores data in vectors to use computer memory efficiently. It maintains indices to locate specific data elements within the vectors. A collection of low-level data management routines moves data to and from the storage vectors.

Depending on the level of reporting requested, the HPPSS will generate as few as two output files per model to over 100 output files per model. Output files for profit and

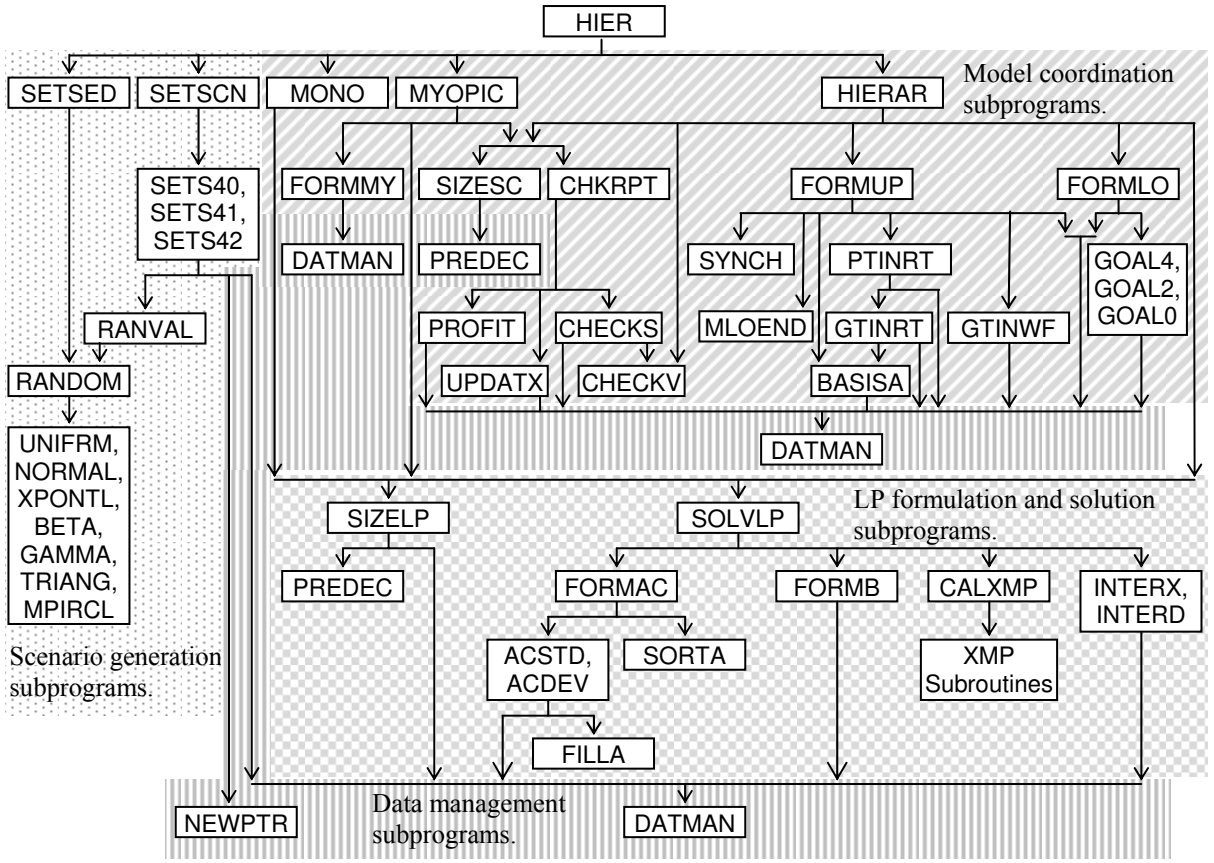

Figure 4: Hierarchical Production Planning System Simulator subprogram structure and calling relationships. (Subprograms are not necessarily shown in the order in which they are called. Subprograms for input/output and error handling have been omitted for space.)

termination status for each scenario are always created for each model. Other output files that can be created for each model contain scenario data, implemented actions over the entire scenario, final plans made each week, intermediate scenarios, intermediate plans, and convergence data. Each data type (e.g., demand, sales, inventory, etc.) is stored in a separate output file. Three additional output files are also created for each run of the HPPSS. These include a log file, a dump of the input data for input verification, and a directory that relates output file names to the various hierarchical models being analysed. The log file can contain data as detailed as the inputs and outputs of each LP if so requested.

The next several sections describe the key subprograms of the HPPSS with an emphasis on technical components that provide the simulator with its power and versatility. In the interest of space, subprograms for input/output, initialization and error handling are neither discussed below nor shown in Fig. 4. Calling relationships between subprograms are shown in Fig. 4.

4.1 Scenario generation subprograms

Using the input initial random number seed, SETSED generates a vector of initial random number seeds, one for each scenario number from one up to the last scenario requested. This generation of initial random number seeds for each scenario is done only on the first call to SETSED and is used to ensure that a given scenario x is the same for any run using the same input initial random number seed, regardless of the first or last scenario numbers for the run. This allows a user to request a minimal amount of output for a run of several scenarios in order to identify scenarios of interest and then to re-run the HPPSS to produce more detailed output for specific scenarios of interest.

SETSED also generates separate random number seeds for each of the parameter distributions for each scenario being analysed. This facilitates the use of common random numbers for parameters that are the same in multiple data sets. For example, hiring cost will be the same in scenario x for all data sets that include hiring as long as both the input initial random number seed and the hiring cost distribution parameters are the same for both data sets. This is true regardless of any other parameters, such as inventory holding cost, which may be included or excluded in a particular data set.

SETSCN and its subroutines generate the random scenarios to analyse using the random number seeds generated by subroutine SETSED for each scenario. The subroutines ensure that the generated scenario parameters are within the required ranges. All scenario parameters generated by these subroutines are stored in scenario parameter vectors.

Function RANDOM and its subprograms are responsible for generating the necessary pseudo random numbers using the uniform, normal, exponential, beta, gamma, triangular, and empirical distributions.

4.2 Model coordination subprograms

MONO coordinates the solution of the monolithic model using the LP formulation and solution subprograms. MONO then reports the implemented actions and the resulting profit.

MYOPIC coordinates the solution of the myopic models. For each myopic planning horizon considered as we move forward in time on a rolling horizon basis, SIZESC calculates the size and map of the myopic model scenario vector which FORMMY subsequently forms. FORMMY starts by transferring data from the overall (monolithic) scenario. Then it uses the solution to the preceding myopic model (if any) to set the initial inventory, backorders, and workforce. It also looks at the end of its horizon with respect to the end of the monolithic horizon and adjusts terminal inventory, backorders, and workforce accordingly. The LP formulation and solution subprograms then form and solve the LP for the current myopic

model. MYOPIC continues forming and solving myopic models using results from each myopic model in the formulation of the next myopic model as we roll forward in time until either steady state is recognized by CHKRPT or the end of a transient problem is reached. MYOPIC then handles the reporting of the implemented actions and the resulting profit.

HIERAR coordinates the solution of the hierarchical models. Each week on a rolling horizon basis, both upper-level and lower-level model scenarios are formed (by FORMUP and FORMLO, respectively) based on the monolithic scenario data, results of the preceding week's models and results of the paired (upper- or lower-level) model for the same week as appropriate for the type of hierarchical model. The LP formulation and solution subprograms then form and solve the LPs. For iterative models, CHECKV determines if plans have stabilized. If not, the simulator iteratively continues to formulate and solve upper-level and lower-level models until stability is achieved or a limit on iterations is reached. If the iteration limit is reached without plan stability being achieved, either the model is terminated or the final plans made are forced to be treated as stable, depending on user options set in the run information file. Once plans have stabilized (either real stability or forced stability), the planning horizon is rolled forward one week and the upper-level and lower-level models are reformulated and solved. This process continues until steady state is reached or a limit on iterations is reached. CHKRPT determines if steady state has been achieved. HIERAR then handles the reporting of the implemented actions and the resulting profit.

SIZESC and CHKRPT are called by both MYOPIC and HIERAR. SIZESC determines the size of the vector needed to store scenario data for intermediate models and sets up a corresponding mapping vector. CHKRPT coordinates the identification of steady-state conditions, updating cumulative plans, and (if so requested) the reporting of final weekly plans and cumulative plans. For steady-state problems, CHECKS compares the solution to a short-term (myopic or lower-level) model to the cumulative results to see if steady state has been reached. The comparison is selective in that CHECKS evaluates whether or not the initial conditions for the next model to be solved in the rolling horizon will be identical to the initial conditions of the corresponding model for the previous cycle (i.e., year). On a data-point-by-data-point basis, the comparisons are carried out by calling CHECKV repeatedly. CHECKV compares a value from the solution to a short-term model to the corresponding value from the cumulative results. Deviation is measured as the absolute value of the difference between the cumulative result and the short-term result when the cumulative result is zero. Otherwise, deviation is measured as the difference expressed as a proportion of the cumulative result. If the deviation encountered between the current values is larger than the previous maximum deviation, CHECKV updates the maximum deviation encountered. Whenever the maximum deviation encountered exceeds the tolerance set by the user in the run information file, subroutine CHECKV sets an indicator to show that steady state has not been reached and that no further comparisons need be done. UPDATX updates cumulative results by transferring results from the solution of the current short-term model into appropriate cumulative result vectors. PROFIT calculates the profit generated by implemented actions for any model.

FORMUP forms the upper-level model for a given horizon. It relies on SYNCH to synchronize the upper-level and lower-level models. Synchronization is necessary because the first few weeks of the lower level's planning horizon may not be in the upper level's planning horizon. For example, if it is the middle of April, the lower level may still be making plans for the last two weeks in April. Depending on the options the user has set, the upper level may also include the last two weeks of April in its planning horizon or it may consider only May and beyond. BASISA obtains the data required for aggregating parameters based on the requested basis for aggregation (i.e., units, sales price, production cost, labour hours, or the sum of production and labour costs). MLOEND converts the end of the lower-level model to

the corresponding time (i.e., month and week) in the monolithic model. PTINRT calculates the initial aggregate inventory and inventory carryover rate (or backorder and backorder retention rate) and puts them into the long-term scenario vector. GTINRT obtains the initial inventory and inventory carryover rate (or backorder and backorder retention rate) and returns the numerators and denominators to complete the necessary calculations with the appropriate weighting for the requested basis. GTINWF obtains the initial workforce from previous results if they exist or from the monolithic scenario otherwise.

FORMLO forms the lower-level model for a given horizon. GOAL4, GOAL2, and GOAL0 establish the appropriate goals, associated penalties for deviations from goals, and corresponding means of calculating the aggregate values for comparison with the goals.

When solving transient problems with hierarchical models, unlike myopic models, it is not appropriate to stop creating and solving models when the first lower-level model to reach the end of the transient is solved. In hierarchical models, solution of a lower-level model that reaches the end of the transient period does not ensure that the starting conditions for the next upper-level model will be the same as the current optimal solution for the next period since plan stability does not mean that upper- and lower-level plans are in agreement.

4.3 LP formulation and solution subprograms

We now turn our attention to the subprograms that implement the master model and serve as a common set of LP formulation and solution routines for the model coordination subprograms (MONO, MYOPIC, and HIERAR). These LP subprograms are the heart of the HPPSS. They allow the user of the HPPSS to focus on the impact of various feedback and control mechanisms rather than the formulation of the various subproblems.

SIZELP determines the size of the LP as measured by the number of rows, columns, and non-zero elements in the constraint coefficient matrix. It also sets pointers to the variables and constraints in the vectors that contain the LP data and results.

SOLVLP coordinates the formulation and solution of the LPs for the various models. FORMAC forms the A matrix and the objective function for the LPs. ACSTD and ACDEV add standard variables and deviation variables, respectively, to the A matrix and objective function. These subprograms call FILLA repeatedly to place the constraint coefficients into the compressed A matrix. SORTA then sorts the A matrix into packed column major order so that each column in the compressed A matrix is sorted by row. This sort facilitates the transfer of the A matrix to the linear programming subprograms. FORMB forms the right-hand-side vector and defines the constraint type for each constraint. CALXMP coordinates the XMP subroutines [19] to solve the current LP. If another set of linear programming subprograms is used to solve the LPs, CALXMP is the only subprogram that will require changes. INTERX and INTERD put the results into vectors similar to scenario vectors so that any calling module can interpret them using the data management subprograms. INTERX translates the LP solution information into the decision variable vector and the reduced cost vector. INTERD translates the dual variables from the LP solution into the dual variable vector.

The compressed A matrix populated by FILLA is an efficient means of storing the constraint coefficients for a sparse A matrix such as occurs in most production planning problems. For example, a production planning problem may easily have hundreds or thousands of constraints with many of the constraints being for specific product groups, items, months, and/or weeks. A specific variable may be included in a very small number of constraints, generally four or less in the master model used by the HPPSS. The compressed A matrix requires a corresponding set of pointers to identify which constraint contains each element of the compressed A matrix. FILLA also sets these pointers.

Table II: Data for example simulation. All distributions are Uniform (Min, Max).

	Demand d_{nm} (units/month)	Sales Price s_{nm} ($/unit)	Production Cost x_{nm} ($/unit)	Labour Required p_{nm} (min./unit)	Holding Cost i_{nm} ($/unit/mo.)
Pattern	Varies by month	Constant	Varies by month	Constant	Constant
Item 1 Min	400	4.00	2.00	4.00	0.06
Item 1 Max	1200	6.00	3.00	5.00	0.08
Item 2 Min	800	2.50	1.50	2.00	0.03
Item 2 Max	2000	3.50	2.00	3.00	0.05

4.4 Data management subprograms

Finally, in this section we turn our attention to a set of low-level subprograms that handle data management. DATMAN and PREDEC are called by many of the above subprograms, but they have not been mentioned explicitly in the subprogram descriptions because they are low-level data management routines that are called frequently to "get" data from the storage vectors or "put" data into the storage vectors.

PREDEC is responsible for mapping the locations of the various arrays contained in a vector. DATMAN is a collection of function subprograms for data management. At the core is a function DATMAN that gets data from or puts data into the storage vectors. Surrounding the core is a set of functions that set up the pointers to the data of interest based on the data indices. The set of indices indicates which array within the storage vector to access. The name and length of the storage vector are sent to DATMAN along with mapping information and instructions to either put data into the array or retrieve data from the array.

5. EXAMPLE

5.1 Example problem description

Consider a simple example for one product group with two items (n = 1, 2). A year consists of 12 months (m = 1, …, 12), each with one "week" and the same pattern repeats year after year. A steady-state plan is desired for production (X_{nm}), sales (S_{nm}) and inventory (I_{nm}). The end items are practically imperishable and can be held for more than a year with no material loss. The end items have stable prices and stable holding costs throughout the year. The components are quite perishable and cannot be held in inventory, so only what will be used is purchased each month at market prices that fluctuate monthly. Backorders are not allowed, as substitutes are available. Space and equipment render the capacity (c) fixed at 30000 minutes (or 500 labour hours) per month. A loyal part-time workforce supplies labour with no attrition. Overtime is not allowed. Direct labour and material costs are incorporated into the production cost. Table II shows the parameters for the data distributions for this example.

The following "master" LP model (which is a subset of the full master model described in Table I) is suitable for this example:

$$\text{Maximize} \sum_{n=1}^{2}\sum_{m=1}^{M}\left(s_{nm}S_{nm} - x_{nm}X_{nm} - i_{nm}I_{nm}\right) - \sum_{m=1}^{M}\left(x_m^- X_m^- + x_m^+ X_m^+\right) - \left(i_M^- I_M^- + i_M^+ I_M^+\right) \qquad (1)$$

Subject to:

$$S_{nm} - X_{nm} - I_{n,m-1} + I_{nm} = 0 \qquad \forall n,m \qquad (2)$$

$$S_{nm} = d_{nm} \qquad \forall n,m \qquad (3)$$

$$\sum_{n=1}^{2} p_{nm} X_{nm} \le c \qquad \forall m \qquad (4)$$

$$I_{n0} = initi_n \qquad \forall n \qquad (5)$$

$$I_{n0} = I_{nM} \qquad \forall n \qquad (6)$$

$$\sum_{n=1}^{2} X_{nm} + X_m^- - X_m^+ = X_m \qquad \forall m \qquad (7)$$

$$\sum_{n=1}^{2} I_{nM} + I_M^- - I_M^+ = I_M \qquad (8)$$

$$I_{nm}, S_{nm}, X_{nm}, X_m^-, X_m^+, I_M^-, I_M^+ \ge 0 \qquad \forall n,m \qquad (9)$$

The objective is to maximize profit (1). The constraints address material balance (2), demand satisfaction (3), capacity (4), and boundary conditions for inventory (5), (6). Constraints on aggregate goals for monthly production (7) or terminal inventory (8) are used by the lower level of hierarchical models only. All decision variables must be nonnegative (9). Not all elements of the master model are used in any single planning model, but the monolithic model and subproblems of the hierarchical models are all subsets of the master model as will be explained below for each model.

For this example, we limit our consideration of hierarchical models to those with two levels in which the upper level is a pure aggregation of the lower level. In other words, the upper level will use aggregated versions of the same variables as the lower level. This limitation is not inherent in the HPPSS as it can consider different variables at the upper- and lower levels. All final decisions for implementation will be made by the lower level with the upper level serving to provide guidance to prevent the lower level from becoming myopic.

5.2 Monolithic model

The monolithic model uses just 5 of the 8 constraints (2, 3, 4, 6 and 9) from the master model. Constraint 6 for steady-state inventory is used exclusively by monolithic steady-state models such as this and ensures that the same sequence of decisions repeats each year after year.

5.3 Single-pass hierarchical models passing production quotas: HProd-*M*

The HProd-*M* models use a single-pass hierarchical modelling approach in which the short-term planning horizon is *M* months. The upper-level model uses aggregate parameters which are simple averages or totals of the corresponding detailed parameters as appropriate.

The upper-level model uses 5 constraints (2, 3, 4, 5 and 9) from the master model. This is almost the same set of constraints as was previously used by the monolithic models. The only difference is that the upper-level model replaces (6) with (5) which constrains only initial inventory ($initi_n$). Also, the upper-level model uses the aggregate parameters while the myopic models use detailed parameters. The upper-level uses a long-term planning horizon of 12 months. The upper-level production plans for the first *M* months are passed to the lower level as production quotas with incentive to meet those quotas.

The lower-level model uses the same 5 constraints (constraints 2, 3, 4, 5 and 9) from the master model as the upper level. The differences are that the planning horizon is only *M* months and the detailed data are used. However, in addition to the constraints already mentioned, the lower-level model adds one additional constraint (7) on the aggregate production goals (X_m) passed down from the upper level and applies penalties (x_m^-, x_m^+) to

negative and positive deviations (X_m^-, X_m^+), respectively.

5.4 Iterative hierarchical models passing terminal inventory targets: ItInv-*M*

The ItInv-*M* models employ a hierarchical modelling approach in which the lower-level planning horizon is *M* months and in which terminal inventory targets are passed from the upper level to the lower level. The upper level in the ItInv-*M* model is identical to the upper level in the HProd-*M* model. However, in comparison to HProd-*M*, in ItInv-*M* the lower level has more freedom over when to schedule production. The lower level in ItInv-*M* differs from the lower level in the HProd-*M* model only in that it replaces the aggregate production goal (7) with a constraint (8) on the aggregate terminal inventory goal (I_M) passed down from the upper level and applies penalties (i_M^-, i_M^+) to negative and positive deviations (I_M^-, I_M^+), respectively. Thus, the lower-level in ItInv-*M* is being given direction as to what state it should be in at the end of its planning horizon rather than instructions as to what to do each month within its planning horizon. Additionally, the ItInv-*M* modelling approach also iterates between levels to adjust the parameters used by the upper level based on planned actions by the lower level and known disaggregated demand within the lower-level planning horizon. Thus, while the upper-level program in HProd-*M* uses constant simple average or total aggregate parameters for each month in its planning horizon, under the ItInv-*M* approach those aggregate parameters are only used for months *M*+1 through the end of the upper-level planning horizon. For the first *M* months of the upper-level planning horizon, the ItInv-*M* modelling approach calculates the aggregate parameters using the latest detailed information available to (or from) the lower level. Thus, if no lower-level plans have been developed yet for a given month within the lower-level planning horizon, the aggregate parameters for that month are based on the weighted average of the individual item parameters where the weights are the individual item demands. However, if lower-level plans have previously been developed for the month, then the aggregate parameters for that month are based on the weighted average of the individual item parameters where the weights are the planned amounts corresponding decision variables for the individual items. For example, the aggregate sales price is calculated by weighting the individual item sales prices by their respective planned sales quantities for the month.

5.5 Simulation study

This simulation study uses the HPPSS to create and evaluate 100 pseudo-randomly-generated scenarios of the example problem described above. We use lower-level models with planning horizons from one through six months (*M* = 1, …, 6). For the upper level of all hierarchical models we use a planning horizon of 12 months. Products are aggregated based on units. To enforce satisfaction of the aggregate production goal (or aggregate terminal inventory goal) if feasible, the simulations impose a large penalty (1,000,000,000) on deviations.

We allow up to 10 cycles through the data to achieve steady state for all modelling approaches. In other words, in a rolling horizon implementation we continue rolling forward until we recognize that we have reached steady state or until we have completed implementation of 10 years, whichever comes first. We allow 5 iterations between levels to reach plan (and hence aggregate parameter) stability in the ItInv-*m* models. If plan stability has not been reached within 5 iterations, then we treat the last (fifth) plan as if it were stable and roll forward in time.

For all scenarios, the demand and production cost for each item and month are drawn from the corresponding uniform probability distribution functions shown in Table II. Also for

all scenarios, the other parameters for each item are drawn from the uniform probability distribution functions shown in Table II but are held fixed for all months.

5.6 Simulation analysis methodology

The measured outcome for each model-scenario pair is the profit realized as a result of implementing the decisions made on a rolling horizon basis over a steady-state year. To overcome biases that could be caused by unusually large or small profits, we analyze each comparison of models using the Wilcoxon Signed Rank Test for a Paired Difference Experiment using the differences between profits for each scenario. Since each model is used to evaluate the same 100 scenarios, we potentially have 100 paired differences for each comparison of models. However, some model-scenario combinations failed to achieve steady state within 10 cycles and therefore could not be used for comparisons. A spot check of model-scenario combinations that did not achieve year-to-year steady state showed that those checked cycled with a periodicity of two to four years.

5.7 Simulation study results

Fig. 5 shows a summary of all simulation results. Fig. 5 shows the average profit for each of the various modelling approaches and planning horizons. It also shows the results of the Wilcoxon Signed Rank Test for comparing adjacent models and the average improvement in profit between those models. We note from Fig. 5 that for short lower-level planning horizons (M = 1, 2, or 3) the HProd-M models perform better than the ItInv-M models. This is because the iterative modifications to the aggregate production costs, holding costs and labour requirements will inevitably be to lower them in the months closest to the end of the lower-level planning horizon as the lower level will always seek the meet the terminal inventory goal as inexpensively as possible. This will lead to a higher planned aggregate production at the upper level and a correspondingly higher terminal aggregate inventory goal for the lower level on the next iteration. For short lower-level planning horizons, this can lead to overproduction of the least expensive item (based on production cost, holding cost, and/or

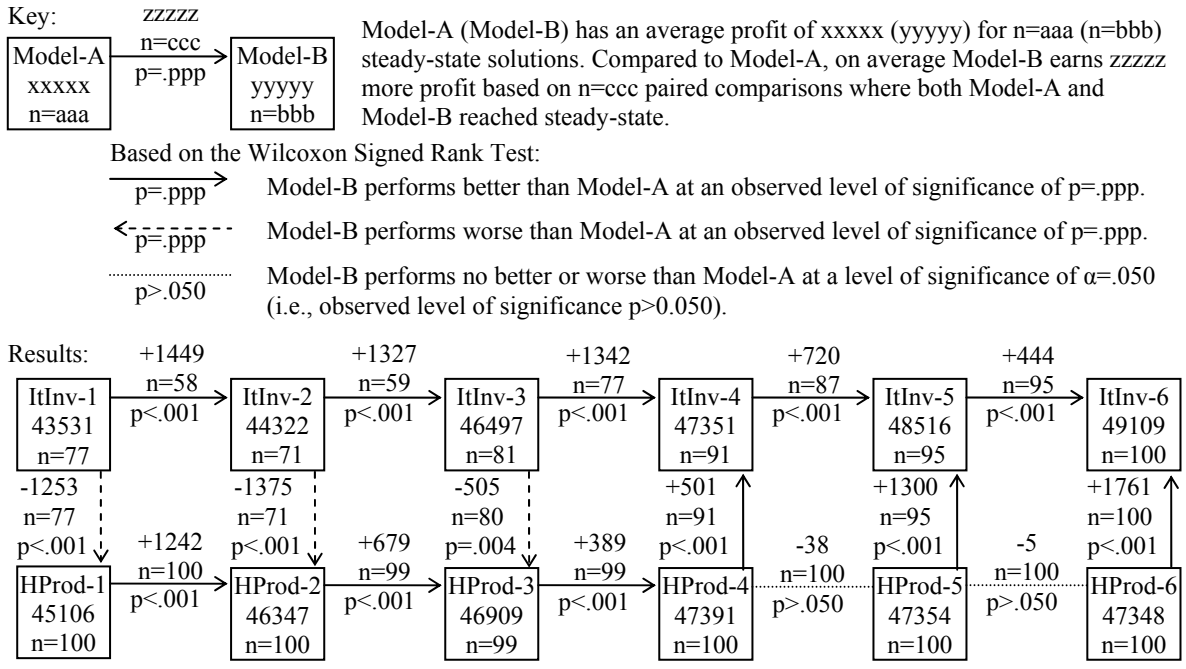

Figure 5: Summary of all simulation test results.

resource requirements) in the first month of the lower-level planning horizon. As the lower-level planning horizon is extended, the impact on the actions implemented based on the first month's decisions diminishes and the benefits of the looser control and the increased information exchange favor the ItInv-*M* models. For each of the models, we also note that performance generally improves as the lower-level planning horizon increases. However, for the HProd-*M* models we note that the improvement disappears after the lower-level planning horizon passes four months. Furthermore, the number of scenarios for which ItInv-*M* matched the monolithic solution rose to 48 for the ItInv-6 model while none of the HProd-*M* models had matched a monolithic solution for a single scenario.

6. CONCLUSIONS

The primary thrust of this research has been to develop a two-level simulation tool for analysing a variety of models and feedback-and-control mechanisms in production planning problems in a more comprehensive fashion than has previously been done. The HPPSS can be used to evaluate the impacts of various information exchanges between the levels in a hierarchical model. Those impacts can be on the outcome of implemented decisions, data requirements and/or computational burden. The comparisons conducted in the example should not be viewed as formal hypothesis tests. However, the results of the example suggest that decision making may improve as: (1) the lower-level planning horizon increases, at least up to some point; (2) control shifts from production quotas to terminal inventory targets for sufficiently long lower-level planning horizons; and (3) upper-level parameters are revised based on lower-level plans. The HPPSS can be used in studies to further support the well-known benefits of longer planning horizons. More significantly, future research with the HPPSS is expected to show the advantages of loose control, the benefits of iterative feedback, and the drawbacks of single-pass mechanisms.

The significance is that a more realistic assessment of "losses" due to system suboptimality induced by various models of the production planning problem can be achieved. This enables better cost/benefit analysis of various models and feedback/control mechanisms. This in turn drives the search for improved feedback/control mechanisms. It will then be possible to formulate planning models that are better equipped to handle the full range of time scales over which the firm must make decisions. This will ultimately lead to better planning and more effective operations.

This research provides an initial framework that can be extended through future research to model the entire production planning and scheduling function of a manufacturing facility operating in a stochastic environment. It will then be possible to formulate models that are equipped to handle the range of time scales over which the firm must make decisions. This will lead to improved operational effectiveness through better planning.

The broader goal of this line of research is to obtain a more thorough understanding of the interaction between the levels of a hierarchical decision-making system over time in order to identify and develop improved hierarchical decision-making techniques. Information exchanges similar to those used within an HPP system are also found in other hierarchical decision-making approaches. Those other hierarchical systems include both computer-based decision support systems and human organizational structures. Thus, the analysis of feedback and control mechanisms may have implications for organizational design and decision-making procedures.

7. ACKNOWLEDGEMENTS

Thanks to the reviewers who made suggestions for significant improvements.

REFERENCES

[1] Hax, A. C.; Meal, H. C. (1975). Hierarchical integration of production planning and scheduling, Geisler, M. A. (Ed.), *TIMS Studies in the Management Sciences. Volume 1. Logistics*, North-Holland/American Elsevier, Amsterdam/New York, 53-69

[2] Miller, T. (2001). *Hierarchical Operations and Supply Chain Planning*, Springer, Berlin

[3] Schneeweiss, C. (2003). *Distributed Decision Making* 2nd ed., Springer, Berlin

[4] Stadtler, H.; Kilger, C. (Eds.) (2008). *Supply Chain Management and Advanced Planning: Concepts, Models, Software, and Case Studies* 4th ed., Springer-Verlag, Berlin

[5] Bitran, G. R.; Tirupati, D. (1993). Hierarchical production planning, Graves, S. C.; Rinnooy Kan, A. H. G.; Zipkin, P. H. (Eds.), *Logistics of Production and Inventory*, North-Holland, Amsterdam, 523-568

[6] Okuda, K. (2001). Hierarchical structure in manufacturing systems – a literature survey, *International Journal of Manufacturing Technology and Management*, Vol. 3, No. 3, 210-224, doi:10.1504/IJMTM.2001.001406

[7] Thomas, L. J.; McClain, J. O. (1993). An overview of production planning, Graves, S. C.; Rinnooy Kan, A. H. G.; Zipkin, P. H. (Eds.), *Logistics of Production and Inventory*, North-Holland, Amsterdam, 523-568

[8] Shapiro, J. F. (1993). Mathematical programming models and methods for production planning and scheduling, Graves, S. C.; Rinnooy Kan, A. H. G.; Zipkin, P. H. (Eds.). *Logistics of Production and Inventory*, North-Holland, Amsterdam, 371-444

[9] Selcuk, B.; Fransoo, J. C.; de Kok, A. G. (2006). The effect of updating lead times on the performance of hierarchical planning systems, *International Journal of Production Economics*, Vol. 104, No. 2, 427-440, doi:10.1016/j.ijpe.2005.04.005

[10] Bitran, G. R.; Hax, A. C. (1977). On the design of hierarchical production planning systems, *Decision Sciences*, Vol. 8, No. 1, 28-55, doi:10.1111/j.1540-5915.1977.tb01066.x

[11] Bitran, G. R.; Haas, E. A.; Hax, A. C. (1981). Hierarchical production planning: a single stage system, *Operations Research*, Vol. 29, No. 4, 717-743, doi:10.1287/opre.29.4.717

[12] Bitran, G. R.; Haas, E. A.; Hax, A. C. (1982). Hierarchical production planning: a two stage system, *Operations Research*, Vol. 30, No. 2, 232-251, doi:10.1287/opre.30.2.232

[13] Graves, S. C. (1982). Using lagrangean techniques to solve hierarchical production planning problems, *Management Science*, Vol. 28, No. 3, 260-275, doi:10.1287/mnsc.28.3.260

[14] Axsater, S.; Jonsson, H. (1984). Aggregation and disaggregation in hierarchical production planning, *European Journal of Operational Research*, Vol. 17, No. 3, 338-350, doi: 10.1016/0377-2217(84)90129-2

[15] Axsater, S. (1986). On the feasibility of aggregate production plans, *Operations Research*, Vol. 34, No. 5, 796-800, doi:10.1287/opre.34.5.796

[16] Lasdon, L. S. (1970). *Optimization Theory for Large Systems*, Macmillan, New York

[17] Gershwin, S. B. (1986). An approach to hierarchical production planning and scheduling, Jackson, R. H. F.; Jones, A. W. T. (Eds.), *Proceedings of the Symposium on Real-Time Optimization in Automated Manufactured Facilities*, NBS Special Publication 724, National Bureau of Standards, Gaithersburg, MD, 1-14

[18] McClain, J. O.; Thomas, J. (1977). Horizon effects in aggregate production planning with seasonal demand, *Management Science*, Vol. 23, No. 7, 728-736, doi:10.1287/mnsc.23.7.728

[19] Marsten, R. E. (1981). The design of the XMP linear programming library, *ACM Transactions on Mathematical Software*, Vol. 7, No. 4, 481-497, doi:10.1145/355972.355976

[20] White, L. R. (2002). *Feedback and control for hierarchical production planning*. Doctoral dissertation, University of Illinois, UMI No. 3070473, ProQuest Information and Learning Company, Ann Arbor, MI