United States
Department of
Agriculture

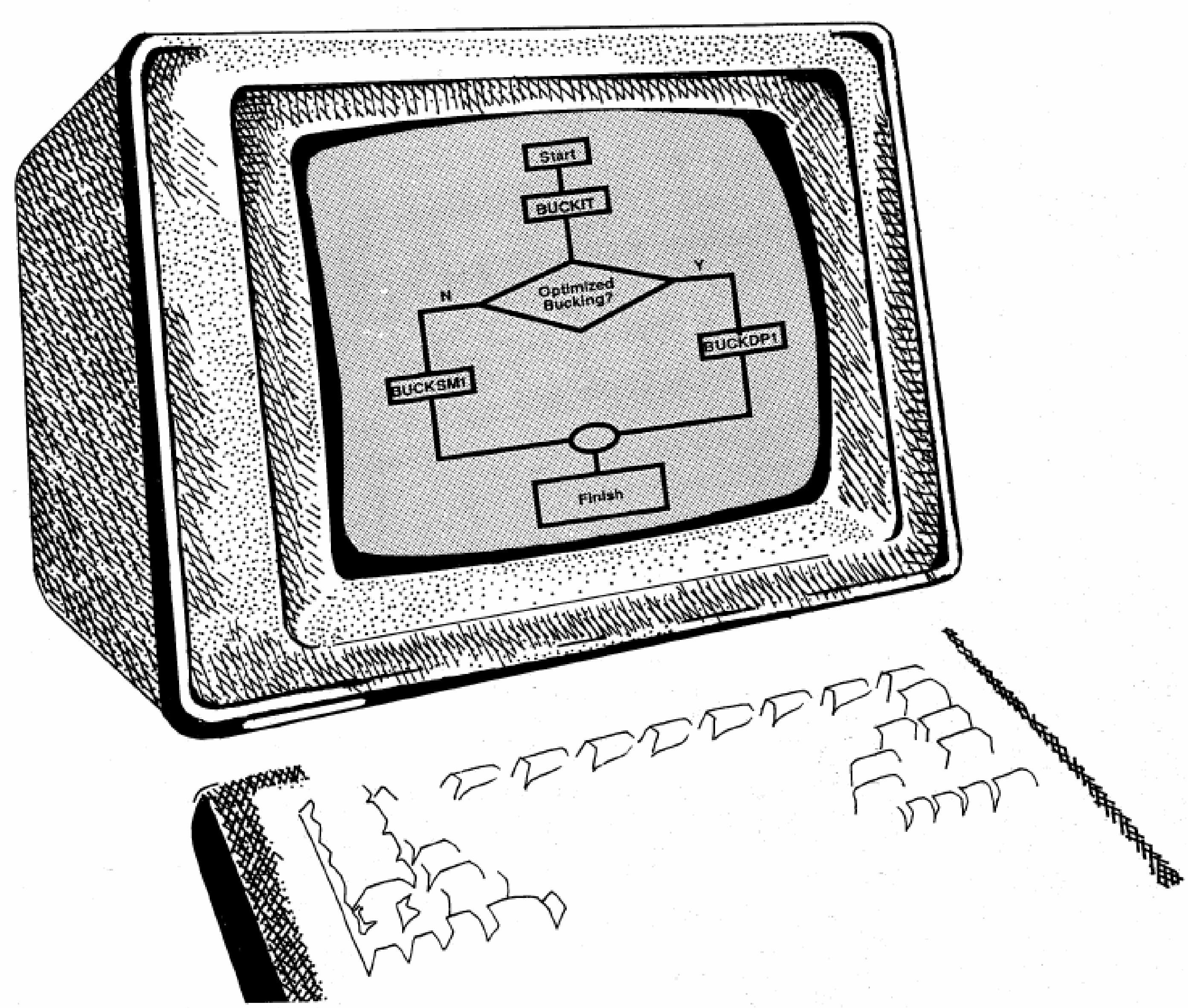
Forest Service

Pacific Northwest
Research Station

General Technical
Report
PNW-GTR-239
April 1989

Tree Value System: Description and Assumptions

David G. Briggs



Author

DAVID G. BRIGGS is associate professor, College of Forest Resources, AR-10, University of Washington, Seattle, Washington 98195. This work was done under a cooperative agreement with the Pacific Northwest Research Station.

TREEVAL is distributed by the Forest Resources Systems Institute (FORS), a nonprofit organization for forestry computer support. A small fee is charged to cover the costs of disks, postage, handling, and telephone support to answer users' questions. For information on ordering a machine-readable copy, write or call:

The Forest Resources Systems Institute
122 Helton Court
Florence, Alabama 35630
(205) 767-0250

Abstract

Briggs, David G. 1989. Tree value system: description and assumptions. Gen. Tech. Rep. PNW-GTR-239. Portland, OR: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station. 24 p.

TREEVAL is a microcomputer model that calculates tree or stand values and volumes based on product prices, manufacturing costs, and predicted product recovery. It was designed as an aid in evaluating management regimes. TREEVAL calculates values in either of two ways, one based on optimized tree bucking using dynamic programming and one simulating the results of user-specified bucking.

Keywords: Tree value, tree volume, product recovery, dynamic programming, bucking, microcomputer.

Contents

Introduction

The Tree Value System (TREEVAL) is a package of computer programs incorporating finished-product considerations into the design and choice of silvicultural regimes.[1] Although TREEVAL was designed primarily to provide value relations useful for making silvicultural decisions, it could also be useful in appraising timber or evaluating bucking practices. TREEVAL consists of three program modules (fig. 1):

1. BUCKIT —An interactive program that guides a user through the various steps of data entry leading to an analysis; includes the choice of using either of the two programs described next.

2. BUCKDP1—A program that converts trees into logs with a dynamic programming-optimization procedure.

3. BUCKSM1—A program similar to BUCKDP1 except that it merely evaluates bucking solutions submitted by the user.

---

[1] The TREEVAL software is in the public domain, and the recipient may not assert any proprietary rights thereto nor represent the software to anyone as other than Government-produced programs.

The TREEVAL software is provided as is without warranty of any kind, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The user assumes all responsibility for the accuracy and suitability of these programs for a specific application.

In no event will the Pacific Northwest Research Station nor the University of Washington College of Forest Resources be liable for any damages, including lost profits, lost savings, or other incidental or consequential damages, arising from the use of or inability to use these programs.

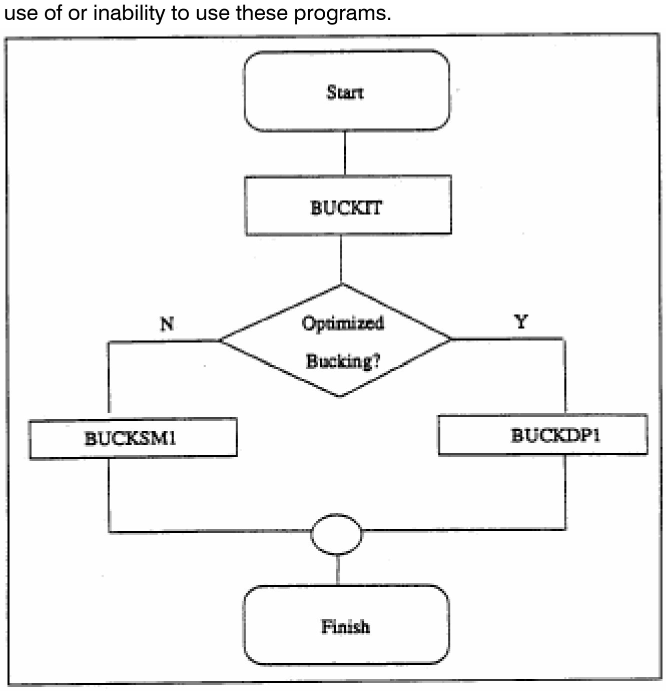


Figure 1—Program components of TREEVAL.

In practice, a user would be unaware of these three programs because the package automatically makes the transition from one program to another.

This report documents the assumptions, procedures, and data requirements of BUCKDP1 and BUCKSM1. See Sachet and others (1989) for additional information on using TREEVAL.

BUCKDP1 and BUCKSM1 are FORTRAN programs that estimate the value of lumber, veneer, pulp, and chips that could be obtained from individual tree stems. Both programs accomplish this by using empirical product recovery relations, prices, and costs to estimate the conversion return of combinations of these products from a tree. The main difference between the programs is that BUCKDP1 uses dynamic programming (DP) to find the optimal combination of log lengths (and their associated end uses) obtainable from a stem, whereas BUCKSM1 evaluates any combination of bucked log lengths with their end-use designations submitted by the user. Thus, BUCKSM1 is useful for evaluating the results of current practice or a proposed bucking rule, and BUCKDP1 estimates what could theoretically be achieved under the same market conditions.

BUCKDP1 : Optimization of Stem Bucking

Dynamic Programming Method for Bucking

Dynamic programming is an approach to problem solving whereby a large, difficult-to-solve mathematical model is decomposed into several smaller, easy-to-solve problems. The DP approach allows the large problem to be decomposed such that once all the smaller problems are solved, the optimal solution to the original large problem is obtained. Each of the smaller problems created is identified with a "stage" of the DP solution procedure; hence, the technique has been used to solve many multistage decision problems. Usually, each of these smaller problems cannot be viewed independently of the others; this is where the DP approach is helpful. Multiple stages often result from a sequence of decisions that must be made over time; for example, a problem requiring a decision over a 1-year planning period could be decomposed into four stages, with each stage requiring an optimal decision over a quarterly time horizon. The stages are the four periods, each spanning a 3-month interval. The decision on what constitutes a stage and what a stage interval is depends on the nature of the problem being solved and the decomposition approach taken by the analyst.

In a physical problem such as log bucking, one decomposition approach is to view the stem as a series of lily pads of constant thickness. Some might prefer to call the lily pads "segments" and the my pad thickness the "segment length." In any event, the lily pads or segments are the stages of the DP problem, and the lily pad thickness or segment length is the stage interval. With few exceptions, the discussion in this paper will use the DP terminology "stage" and "stage interval."

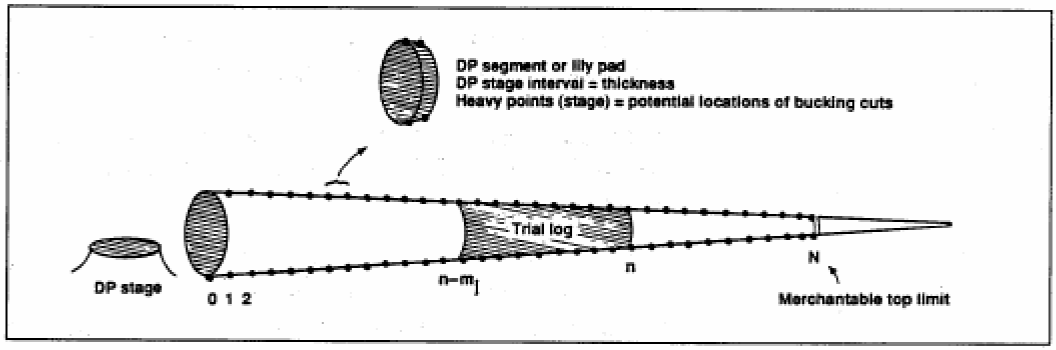


Figure 2—Dynamic programming applied to bucking. The stem is series of contiguous lily pads. Each trial log length is a series of contiguous lily pads.

In the DP approach to bucking, the stages are potential locations where bucking cuts could be made (fig. 2). Diameter inside bark (d.i.b.) at each stage is estimated from a taper equation. Procedures within BUCKDP1 use empirical relations for pulp yield and for lumber and veneer grade yields and residues; appropriate prices and manufacturing costs are included. The user can describe a single process associated with using a log in each of the following markets: lumber, veneer, pulp, and chips. Included in each process description are the desired log lengths with trim, the range of acceptable log diameters, and the recovery, price, .and cost data. BUCKDP1 produces a bucking solution showing the optimal total conversion return for the stem along with the specific location of each bucking cut, the corresponding log descriptions, and their market designations.

To illustrate the development of the DP technique for stem bucking, consider a tree stem (fig. 2) that is to be bucked between the stump and merchantable height limit. Some upper height limit less than the total height of the tree will always occur because the bole diameter eventually becomes too small to yield logs acceptable in any market. Also, the top may break off during felling or the upper stem may fork into unusable sections.

The markets require logs of specific lengths including trim allowance. Sometimes, different markets compete for the same length. In addition, the markets may have upper as well as lower limits for acceptable log diameters. These diameter limits are normally specified for the small end of the log.

The log lengths and the potentially usable stem length are divided into stages with a constant stage interval. The stage interval (segment length) is chosen such that all the log lengths are an integer that is a multiple of this value. The DP thus views any log length as a series of contiguous equal-length stages. Numbers that lead to non-terminating fractions might sometimes be chosen. Recommendations are discussed in the section, "Use of BUCKDP1."

The DP approach to log bucking can be expressed mathematically as follows. Let:

j = 1,...,J refer to the log lengths;
n = 1,...,N refer to the DP stages; .
m_j = number of contiguous stages comprising the j-th log length;
$CR(n, n - m_j)$ = conversion return of a log spanning stages $(n, n-m_j)$ based on log dimensions, log scale, product grade recovery, product prices, and manufacturing costs; , .
f_n = maximum conversion return of stages 1 ,...,n if stage n is part of the last log bucked; and
F_n = maximum conversion return of logs bucked from stages 1,...,n.

The dynamic programming recursion is:

$$F_n = \max (f_n, F_n - 1) , \qquad (1)$$

where,

$$f_n = \max [CR(n, n - m_j) + F_{n-m_j}] ,$$

and the boundary condition is:

$$F_n = f_n = 0 \text{ for } n = 1,\ldots,\min (m_j) .$$

The approach may be explained as follows. Assume a bucker starts at one end of the stem and moves toward the other end. The bucker is assumed to move from the stump toward the top, although the direction could be reversed. The bucker stops at the end of each stage that is the Potential location of the small end of a bucked log. Assume that the bucker is presently at point n. The bucker evaluates the various log-to-market alternatives with each of the j trial logs terminating at this small-end location, and combines the result from each trial log with the value of the best solution already obtained between the base of the stem and the large end of the trial log at stage $n - m_j$. Selecting the stage interval such that all log lengths and the stem are integer multiples of the stage interval guarantees that any log will always have each end coincide with a stage. The bucker notes the value of the best of these evaluations and compares this result with the value of the best solution for the whole stem that was recorded at stage n - 1. If the best evaluation found at stage n is greater than the value at stage n -1, the bucker records the log length and market allocation that is the bucking decision at stage n. Otherwise, a solution at a previous stage was better, so "do not cut" is recorded at stage n. Following this procedure, the bucker eventually reaches the merchantable height limit. Then the bucker scans back through the records to identify the optimal pattern and allocation for the whole stem.

This description of log bucking using DP applies to either woods-length (long) log bucking or to mill-length log bucking. The problem of optimally bucking a tree into woods logs and subsequently bucking these into mill logs, although conceptually similar, involves a more complex formulation and is not treated by TREEVAL. In TREEVAL, the DP formulation is applied in the sense of mill-length log bucking.

Figure 3 presents an overview of the procedures used to incorporate this process into BUCKDP1. Subsequent sections will discuss these steps in greater detail and the routines in BUCKDP1 that implement them.

Several researchers have developed DP models similar to the preceding description to address the general problem of stem bucking (Briggs 1977, 1980; Deadman and Goulding 1979; Faaland and Briggs 1984; Geerts 1979; Gluck and Koch 1973; Pnevmaticos and Mann 1972). Nasberg (1985) provides a comprehensive review of these models and Lembersky and Chi (1984) present an interactive visual graphics form for training buckers.

The basic approach of these authors has been essentially the same; divide the stem lengthwise into lily pads serving as DP stages. Gluck and Koch (1973) tried an approach using the cut number as DP stages, but this is less efficient computationally. The main differences among the authors are the details of incorporating log specifications, log scaling, markets, product recoveries, and stem defects into the analysis. A general progression toward more realistic and sophisticated analyses has occurred over time.

Figure 3—Computational overview of BUCKDP1.

1. Read the log lengths to be considered, list them from shortest to longest, and then read the DP stage interval
2. Read the marketing alternatives for each log length and the range of acceptable diameters in each market
3. Read the product recovery data, prices for products and residues, and manufacturing costs
4. Read the taper equation coefficients
5. For each tree (or representative of a tree class)
6. Read the tree description: d.b.h., total height, stump height, and number of trees
7. Calculate and store the inside bark diameter profile by using the taper equation. Estimate diameter at each DP stage
8. For each DP stage
9. For each alternative log length
10. Check whether current length is feasible betWeen stump and current stage
    Yes; go to step 11
    No; go to step 24 (remember lengths are listed in ascending order)
11. Calculate actual dimensions of the log
12. For each market (LUMBER, VENEER, PULP, CHIPS)
13. Check if the log length is accepted in the current market
    Yes; go to step 14
    No; go to step 20
14. Check if the small-end diameter of the log is within the market's acceptable range
    Yes, go to step 15
    No; go to step 20 15. Calculate cubic-foot volume of the log
16. Estimate product recovery from the log and, if lumber or veneer, also estimate grade and residue volumes
17. Apply prices to product and residue volumes to estimate total log revenue
18. Apply costs to get conversion return

19. Check if the conversion return from the current log is better in the current market than in previously evaluated markets
    - Yes; store current market as best
    - No; retain previous market as best
20. Next market
21. Combine the conversion return for the best market use for the current log with the optimal conversion return for the region between the stump and butt end of the current log
22. Check if the optimal conversion return for the stem with the current log exceeds that of the previous best solution among other logs evaluated at the current stage
    - Yes; store the current log as the best to buck at the current stage
    - No; retain the previous best log as the best to buck at the current stage
23. Next log length.
24. Check if the optimal conversion return for the current stage, based on cutting a log at that stage, is at least as good as some prior-stage optimal conversion return
    - Yes; retain the current log as the best stem solution through the current stage
    - No; change the decision to do not cut any log at the current stage; all options at the current stage are inferior to a prior solution
25. Next stage
26. Print detailed bucking pattern (optional) for current tree and accumulate for stand summary
27. Next tree
28. Print stand summary (optional) or write summary to a file (optional), or both
29. End

Assumptions in BUCKDP1

BUCKDP1, as developed here, is primarily a tool for predicting the product value of stands grown under various silvicultural regimes and is derived from previous efforts (Briggs 1977, 1980; Faaland and Briggs 1984). The stands of interest often represent hypothetical or modeled rather than actual stands. Because the objectives of this research were primarily for long-term strategic planning of silvicultural decisions about growing trees rather than short-term or instantaneous decisions of how to buck a tree length or long log on a deck, BUCKDP1 has some assumptions that differ from previous work.

1. The d.i.b. at each DP stage is estimated from a taper equation instead of from actual measurements supplied by an electronic scanner system at a mill.

2. Shape abnormalities such as sweep and crook and internal rot are not considered. Because the intent of the model is to forecast product value from stands grown under intensive management, it is assumed that trees developing these defects will be a minor stand component during commercial thinning and at harvest. Similarly, no assumptions are made about out-of-round cross sections of stems; the empirical product-recovery study results used are based on many trees and are assumed to take cross-sectional variability into account.

3. All product-recovery and cost relations were assumed to be functions of small-end log diameter of the general form:

$$Y = a_1 + a_2D + a_3D^2 + a_4/D + a_5/D^2 , \qquad (2)$$

where

Y = volume recovery factor, grade yield percentage, or cost per unit of volume;
D = small-end scaling diameter in inches; and
a_i = coefficients dependent on species and product.

The units of measure in specific uses of this equation are explained below. By entering zeros for combinations of ai, many commonly used recovery and cost relations can be modeled. The choice of this macro form was based on existing product recovery studies, most of which have used only diameter as an independent variable. This approach may be inadequate for addressing quality considerations that will emerge as forests undergo the transition from natural to intensively managed stands. The development of more sophisticated relations is a subject for ongoing and future research. New relations will be easy to introduce after they have been defined.

4. Manufacturing costs predicted by equation (2) produce a cost estimate in dollars per cunit (100 cubic feet) of log volume.

5. Scaling of logs and use of recovery information follow practices used by the Timber Quality Research Project, Pacific Northwest Research Station.[2] A data file containing recovery coefficients usable in equation (2) for second-growth Douglas-fir *(Pseudotsuga menziesii* (Mirb.) Franco var. *menziesi)*_ green veneer and surfaced-dry dimension lumber is provided with the TREEVAL system. The user may substitute coefficients from other recovery studies but should be careful that the coefficients are consistent with the following conventions:

   a. Logs for lumber manufacture have their estimated diameters rounded to the nearest inch, and length is converted to nominal. Lumber is priced by grade and length. Price premiums can be given for these two characteristics but not for cross-sectional dimensions. Lengths for saw logs are assumed to be mill lengths as opposed to woods lengths. The recovery-equation data are assumed to be valid over a small-end diameter range of 6 to 25 inches. Any log less than 6 inches at the small-end is assumed to have the recovery and grade yield of a 6-inch log. Any log exceeding 25 inches is similarly assumed to have the recovery and grade yield of a 25-inch log.

---

[2] Ernst, Susan A. 1983. Study plan for multi-product recovery of coast Douglas-fir in western Washington. Project 01-10. 16 p. On file with: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station, P.O. Box 3890, Portland, OR 97208-3890.

b. Because saw-log volumes are calculated from the nominal log length, excluding trim, the residue element resulting from lumber trim is incompletely represented in the recovery relations. A separate step calculates the cubic volume represented by the trim ends and prices it according to the **first** price entered in the sawmill-residue price list. Thus, if the user wants the material priced as chip residue, chip residue should be entered first.

c. Peelers for veneer are initially 17 feel long and are divided into two blocks for recovery estimation based on actual block dimensions. The recovery-equation data are assumed to be valid over a small-end diameter range of 10 to 26 inches. As explained above for lumber, any veneer block having a diameter below or above these limits is assumed to have the recovery and grade yield of the appropriate end point of this range.

d. Pulp and chip calculations are based on actual log dimensions.

6. All log volumes are estimated in cubic feet. Cubic volume of logs with the large end within 4.0 feet of the stump is calculated with Bruce's butt-log formula (Bruce 1982). When the large end is above this point, cubic volume is calculated with Smalian;s formula.

7. Log grades are not explicitly considered. Previous research with old growth found that log grading seemed to have only a 5- to 10-percent effect on predicted stem value (Briggs 1980). In the common log-grading systems used in the Pacific Northwest, second-growth logs usually fall in one or two log-grade categories, and log grading does not improve estimates of the value of second-growth logs.[3]

When using BUCKDP1, the user must ensure .that units of measurement in the recovery equation (equation 2) and the prices entered are consistent and that the markets (LUMBER, VENEER, PULP, CHIPS) are properly spelled. Presently, BUCKDP1 operates as follows:

1. Lumber recovery is estimated in boardfeet (BF) mill tally, prices in $/thousand boardfeet (MBF), and the program converts BF to MBF. The coefficients in the default recovery file are for surfaced-dry dimension lumber, so prices and lumber manufacturing costs should be expressed on this basis. The user can enter data for another-basis, such as rough green, as long as the preceding conventions are followed.

2. Veneer recovery is estimated in square feet. The default data supplied in the recovery file is on -a green 3/8-inch basis; The program converts recovered veneer in square feet to thousands of square feet (MSF). The price in $/MSF must be expressed consistently with, the recovery data (that is, 3/8-inch basis with the default data).

---

[3] Personal communication, Thomas D. Fahey, research forester, U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station, P.O. Box 3890, Portland, OR 97208-3890.

3. Recovery of residues is estimated in cubic feet of solid wood equivalent (s.w.e.). Prices for residues must be given in $/cubic foot (s.w.e.).

4. There is no provision in BUCKOP1 fora recovery factor to convert solid logs to the product, CHIPS. Thus, the price for chips, commonly $/unit (200 cubic feet of chips), $/bone dry unit (BDU) (2,400 pounds or 1088.6 kilograms), or $/bone dry tonne (BDT) (1000 kilograms or 2204.6 pounds), must be converted to a solid roundwood basis to use this product option. This option refers to processing whole logs into chips and is distinct from chippable residue resulting from the conversion of logs to veneer or lumber. Manufacturing residues of veneer and lumber are priced and tallied as part of the revenue for LUMBER or VENEER logs.

5. Pulp can be expressed in more than one way depending on the desires of the user. The user must enter data values that yield consistent units of measurement and that are compatible with log-volume scaling in cubic feet. The following examples illustrate procedures for two options.

   a. Pulp pricing—The cubic-foot log volume is combined with a pulp-process recovery factor and a pulp price to obtain a pulp-revenue estimate. Cost of pulping must be expressed in units consistent with the result of the revenue calculation. Although this is seemingly straightforward, the user must consider several conversion-factor problems.

   First, a pulp-process recovery factor of, say, 0.45 generally means that 45 percent of the ovendry weight of the input log becomes pulp. If a pulp log weighs 60 pounds per cubic foot and if it has a 50-percent moisture content on a total weight basis,[4] then each cubic foot of the log yields 30 pounds of water and 30 pounds of ovendry wood, which in turn yields 13.5 pounds of ovendry pulp and 16.5 pounds of ovendry residue. The pulp-process recovery of 0.45 translates to a value of 13.5 pounds of pulp per cubic foot of log input. This translated value, dependent on specific gravity and moisture content, is compatible with the cubic log volume scaled by BUCKOP1 and thus should be the value used in the pulp-recovery data file.

   A second difficulty arises from the various ways of reporting pulp prices. Although pulp prices are usually given in dollars per ton, the ton can be in English or metric measure. Either way, the price may reflect ovendry (0 percent) or air-dry (about 7 percent) moisture content on a total-weight basis. To be consistent with the result of multiplying log volume in cubic feet by the translated recovery factor (ovendry pounds of pulp per cubic foot of log), the pulp price must be translated and entered in dollars per ovendry pound of pulp.

---

[4] The pulp and paper industry generally expresses moisture content on a total-weight basis in contrast to the lumber and plywood industry, which generally expresses moisture content on an ovendry-weight basis.

Pulp manufacturing costs are generally reported in dollars per ton, just as prices are. The user should convert the estimate of manufacturing cost to dollars per pound of ovendry pulp, subtract it from the pulp price, and enter the net figure. The procedure of subroutine COST could be used except that the cost equation in this subroutine uses coefficients that produce a cost estimate in cunits of log-input volume. A pulping cost estimate can be translated to a log-input volume basis, but this involves additional manipulation of conversion factors.

b. Pulp roundwood pricing-The cubic-foot log volume must be combined with the price per cubic foot that a mill pays for delivered pulpwood. This easily can be accommodated in three steps:

(1) Set the recovery factor in the pulp-recovery file to 1.0.

(2) Convert the pulpwood price to dollars per cubic foot of log.

(3) Set the pulp-process manufacturing cost coefficients to zero.

BUCKDP1 Program Organization

Figure 4 presents a schematic of the relations among the modules that comprise BUCKDP1. The following sections describe the modules.

Main program BCKDP1— The main program BCKDP1 opens the tree data file containing the taper equation code, corresponding coefficients, and tree descriptions. It also opens the output files DPTABLE and DP-REG.PRN that will store the stand summary. It initializes variables used for calculating summary statistics for the stand and for the current tree class. Then, it invokes subroutine INFO, which reads all data in the market and product-recovery data files. Next, it reads the taper equation information from the tree file and performs a loop that reads data on a new tree class, accumulates class statistics, and invokes subroutines SCANER, DPRO, and RESULT. These subroutines create the stage-by-stage d.i.b. profile, perform the DP

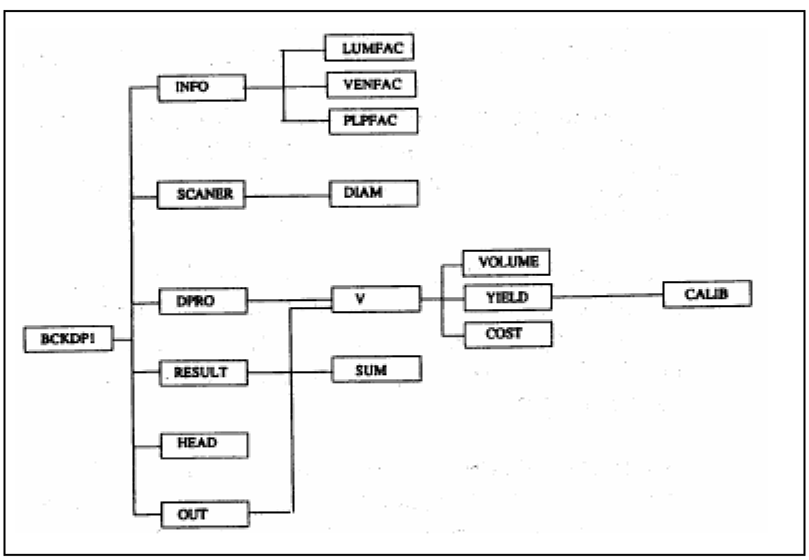


Figure 4—Program modules in BUCKDP1.

calculations, and search the stage-by-stage solutions to identify the logs and corresponding market allocations that comprise the optimal solution for the current tree class. Based on the output option chosen, BCKDP1 also invokes routines producing the desired output.

Subroutine INFO—This subroutine contains two major sections. The first section opens the data file containing all log and market specification data and reads in the segment length used to define the DP-stage interval, the list of log lengths and associated trim, the markets and corresponding minimum and maximum accepted diameters, and the pointers (0,1 codes) that indicate which of the log lengths accepted are (1) or are not accepted (0) by each market. It also reads from this file the prices for products and residues and the coefficients used in the manufacturing-cost equation for each market.

INFO also opens the file containing the coefficients required by the product-recovery macroequation (equation 2) used for lumber, veneer, and pulp. Separate subroutines LUMFAC, VENFAC, and PLPFAC read these coefficients.

After the data have been read, these input files are closed, and the second section provides an echo of the market-file information. An echo is not provided for the product _recovery coefficients.

All the information provided from these files, is entered or modified by executing BUCKIT (Sachet and others 1989), which leads the user through data entry and editing.

Subroutine LUMFAC—This subroutine reads coefficients for lumber recovery used in equation (2). The first set of coefficients predict lumber recovery factor (LRF) in board feet per cubic foot of log. Second, the number of residue categories is read; for each residue, its name (for example, NONCHIP, CHIP) and the corresponding equation coefficients to predict the percentage of log cubic volume that becomes the particular residue are read. The number of lumber grades are read last, and for each grade, its name (for example, SEL STR, UTILITY) and the corresponding coefficients for the equation to predict the fraction of lumber by grade are read.

The documentation for subroutine YIELD presents details on using these data. The residue and product names (CHIP, NONCHIP, SEL STR, and so forth) are mainly used to help users identify information in files and are not critical descriptors in the program.

Subroutine VENFAC—This subroutine performs the same actions for veneer as are performed by LUMFAC for lumber recovery. First, VENFAC reads coefficients for veneer recovery factor (VRF) in square feet of veneer per cubic foot of log, then coefficients for residue percentage, and lastly coefficients to predict the fraction of veneer by grade.

Subroutine PLPFAC—This subroutine reads in the recovery factor for a pulp process. This factor is assumed to be constant regardless of log diameter and must be expressed as some unit of pulp (or paper) obtained per cubic foot of log volume.

Subroutine SCANER—Starting at the stump, SCANER calculates the height in the tree to each DP stage and uses the chosen taper equation (FUNCTION DIAM) to estimate the corresponding d.i.b. These d.i.b.'s are stored and subsequently used as a table when the diameters of a log are needed during the DP calculations. SCANER terminates when the d.i.b. falls below the smallest diameter acceptable in any of the markets. The d.i.b.'s are stored as radii as if the central axis of the tree were zero. Thus, an upper or positive radius coordinate and a lower or negative radius coordinate are used.

SCANER permits a list of up to 600 d.i.b.'s. The program will print a message and stop if an analysis produces a situation violating this dimension. Because analyses will commonly use a DP stage interval of 0.25, 0.33, or 0.5 feet (4, 3, or 2 intervals per foot), this dimension should be more than adequate to accommodate the merchantable bole of any second-growth tree.

Function DIAM—This function estimates the inside bark diameter (DI) at a given height (HI) of a tree of specified diameter at breast height (d.b.h [DBH in this function]) and total height (HT) depending on the user's choice of taper equation.

Presently, the user can select among three different taper equations. The first two require the user to supply coefficients for the appropriate species. These coefficients and the selection code are read from the tree file by the main program BCKDP1. The third equation is specific to Douglas-fir, and the coefficients are permanently included in FUNCTION DIAM. In this case, the data file must have the selection code and fictitious or zero coefficients. The equation choices are:

$$(DI/D)^2 = a_1 + \_(HI/HT) + a_3(HI1HT)^2 \qquad (3)$$

(from Kozak and others '1969),

$$(DI/D)^2 = a_1X^{1.5} + a_2(X^{1.5} - X^3)D + a_3(X^{1.5} - X^3)HT + a_4(X^{1.5} - X^{32})D*HT + a_5(X^{1.5} - X^{32})HT^{0.5} + a_6(X^{1.5} - X^{40})HT^2 \qquad (4)$$

(from Bruce and others 1968), and

$$(DI/D)^2 = X^{1.5} + (\_.1661 + 5.8372/HT)(X^{1.5} - X^{32}) + (.11906D^2 - 6.9988/HT)(X^{1.5} - X^2) + (12.2416/D - 1017.43/HT^2 - 295.762/D/HT)(X^{1.5} - X^3) \qquad (5)$$

(from Bruce[5]);

where
D = d.b.h. in inches,
HT = total height in feet,
DI = d.i.b. predicted at height HI, and,
X = relative height (HT - HI)/(HT - 4.5). For equation (5), if HI < 4.5 feet, DI is multiplied by 1.082 to adjust for underprediction.

---

[5] Unpublished data on file with: D. Bruce, Mensurationist, USDA Forest Service, Forestry Sciences Laboratory, P.O. Box 3890, Portland, OR 97208-3890.

Equation (5) was developed specifically for coast Douglas-fir, whereas equations (3) and (4) have been applied to more than one species. The user should consult the original sources of these equations, Smith and Kozak[6] and Gupta (1988), for additional background and coefficients for various species.

Subroutine DPRQ—This subroutine implements the DP algorithm and uses subroutine V and associated routines to evaluate the conversion return of each candidate log at a given stage. This subroutine can be described briefly as follows. Imagine the bucker starting at the stump of a felled tree that is now represented as a series of DP stages for which the d.i.b. is known. Each stage is considered as a potential site for bucking the tree and is coincident with the location of the small end of a log. This is why the DP-stage interval was defined such that all the log lengths are an integer multiple of it; all logs will have both ends at a DP stage. If the log lengths were listed in. order from shortest to longest, then none of the log lengths are feasible until the bucker moves away from the stump by the number of stages equivalent to the length of the first (shortest) log in the list; this stage is defined by the variable LIMIT. For all stages before this location, no usable logs can be cut; hence; there is no value. The first loop in DPRO recognizes this fact, which is called the "boundary condition" in DP terminology. For these stages where no logs are feasible, this loop initializes array F, which stores the optimal conversion return through the current stage, and initializes array JP, which stores the code of the log we would buck at the current stage. Because no logs are feasible to buck until the stage corresponding to LIMIT is reached, F is set to a return of zero and JP is set to minus 1 to indicate no log size can be cut

The second loop examines the remaining stages where at least one of the lengths from the log list is potentially feasible. Assume that the model is presently at stage K. After initializing a variable (SMALLF), which keeps. track of the best value obtained from the various lengths that might be cut at stage K, another loop is invoked.

This third loop systematically examines each of the lengths in the log list. It determines if the current log length is longer than the usable stem region between the stump and stage K. Because the lengths in the log list are ordered from shortest to longest, the third loop discontinues log calculations for stage K as soon as a length that is too long is encountered. If the current log length is potentially usable, the stage number (K1) corresponding to its large end is identified. [7] Next, subroutine V evaluates the trial log, checking its use among the various markets and calculating product value and cost of processing to get a conversion return for each.

---

[6] Smith, J. Harry G.; Kozak, Antal. 1971. Further analysis of form and taper of young Douglas-Fir, western hemlock, western red cedar and silver fir in the University of British Columbia Research Forest. Paper presented at Northwest Scientific Association Annual Meeting, University of Idaho, April 16-17. 8p.

[7] Variables K and K1 in program BUCKDP1 refer to locations n and $n\text{-}m_i$ in figure 2.

Subroutine V returns to DPAO the highest conversion return found among the markets. This result is combined with the optimal conversion return previously found for the portion of the stem between the stump and butt (large) end of the current log; that is, the optimal value at stage K1. This combined total conversion return for the solution between the stump and current stage is compared to the best value found for other log lengths that could be cut at stage K; either a previously analyzed log is retained as being part of a better solution or the solution with the current log replaces it. SMALLF stores the total conversion return, and JP stores the code of the log length.

When all lengths have been evaluated at stage K, a final comparison determines whether the optimal conversion return obtainable if a cut is made at stage K is inferior to the optimal solution recorded at stage K-1. If this is so, a log should never be cut from stage K. Thus, the optimal value is carried forward from the previous stage in array F, and the length code in array JP is changed to indicate do not cut (-1).

After these checks are complete, the DP algorithm is repeated for the next stage. The calculations for the last stage are eventually completed, and information on the optimal solution is stored in lists F and JP. Control returns to BCKDP1, and subroutine AESULT locates the pattern associated with the optimal value.

Subroutine V-This subroutine estimates the highest conversion return for the current log length being evaluated by subroutine DPAO. The subroutine first calculates the d.i.b.'s of the log and then calculates the length and height position of the log along the bole. Next, the log is examined for use in each of the markets. When the market pointers indicate that the length is accepted by a market, the small-end d.i.b. is checked to determine if it is within the diameter range accepted by the market. If the log is acceptable, the revenue from products and associated processing costs are estimated, and the conversion return is compared with the best previous use for this log, always retaining in the program the market yielding the highest conversion return. When all markets that can use the log have been examined, the highest conversion return is returned to DPAO. The details of how each market is handled are described below.

If the log is acceptable for lumber manufacture the diameters (rounded to the nearest inch), the nominal length (that is, total length minus trim), and the height of the large end of the log are submitted to function VOLUME, which estimates the cubic-foot volume. The reason for submitting the height to the large end of the log is that this value acts as a switch determining whether Bruce's (1982) butt-log formula or Smalian's formula is used. The next step is to extract the list of lumber grade prices corresponding to the nominal length. Subroutine YIELD calculates the revenue obtained from lumber and residues. The residue yields in the recovery database were based on procedures whereby the log cubic volume was calculated from its nominal length. The procedures used in developing recovery equations thus underestimate chippable residues by the amount of trim ends. The volume and revenue from chipped trim ends is estimated by assuming that this residue price is first in the residue price list. Finally, the cost of processing the log into lumber is deducted. Manufacturing cost is expressed in cubic volume of log input, not lumber tally.

For veneer manufacture, peeler logs are assumed to be 17 feet long. The length is divided into two 8.5-foot blocks by subroutine V. The end diameters, length, and height position of each block in the tree are used by function VOLUME to estimate cubic volume based on either Bruce's butt log or Smalian's formula. For estimating cubic volume, the actual diameters and length of the block are used. The block dimensions are submitted to subroutine YIELD, which estimates the total revenue obtained from veneer and residues. After the revenue is calculated, the cost of processing the block is estimated and deducted to yield the conversion return. As for lumber cost, manufacturing cost is based on block volume rather than veneer tally. The results for the two blocks are combined to give a total for the original peeler log.

Pulp manufacture is based on the actual log dimensions converted to cubic feet in function VOLUME by using either Bruce's butt-log formula or Smalian's formula. Recovery is based on a pulp-process recovery factor that converts a cubic foot of roundwood log to pulp or paper. The user must supply a price per unit of product that is consistent with the recovery factor and a similarly consistent manufacturing cost expressed in cubic feet of roundwood log input. The default file contains a pulp recovery factor of 1.0 to reflect the pulp roundwood pricing option discussed above in the section, "Assumptions in BUCKDP1."

Converting a whole log into chips is similar to converting to pulp except that no recovery factor is used. That is, a log for chips is handled by the model as if it were sold in roundwood form.

Function VOLUME—This function estimates the cubic-foot volume of logs. It uses Bruce's (1982) butt-log formula whenever the large end of the log is within 4.0 feet of the stump, and it uses Smalian's formula when the large end is above this point. Experimenting with these formulas for several tree sizes suggests that switching at about this height minimizes discontinuities between these formulas.

Subroutine YIELD—This subroutine estimates recovery, grade yield, and residues from logs processed into lumber or veneer and combines this information with prices to estimate total revenue. Because the recovery equations are based on empirical regressions, a danger exists in extrapolating beyond the range of saw-log and veneer-block diameters used to develop the equations. The first step is to identify the range of valid predictions: 6-25 inches d.i.b. small end for saw logs and 10-26 inches d.i.b. small end for veneer blocks. If a log is outside these limits, the recovery calculations are based on the end point of the range. These limits are included in this subroutine and are not entered as data by users.

Next, the lumber recovery factor (LRF = board feet of lumber per cubic foot of log input) or veneer recovery factor (VRF = square feet of veneer per cubic foot of block input) is estimated. These factors, when multiplied by the log volume and divided by 1,000; estimate the total MBF or MSF produced from the log. For veneer, the calculations make no assumptions on thickness (that is, 1/8 inch, 3/8 inch, and so forth); the user must ensure that the recovery and price bases are consistent. If the default file is used, the basis is 3/8 inch. Because veneer recovery is based on actual raw green sheet dimensions but veneer prices are quoted on a 4- by 8-foot nominal basis, the recovery equation overestimates the volume of veneer actually paid for. The initial estimate of green veneer is reduced by multiplying by 0.88 to reflect a 12-percent differential between actual and nominal green veneer sheet size (Fahey 1987).

The next step is to estimate the grade-yield fractions from the log. Because the regression estimates of these fractions may not sum to exactly 1.00, they are monitored and used by function CALIB to create a correction factor. These adjusted grade-yield fractions are multiplied by the total MBF or MSF of product and by the corresponding prices per MBF or MSF and are summed.

When this is completed, the percentage of the log cubic volume in each residue class is calculated, converted to a fraction, multiplied by the log volume, and then multiplied by the price per cubic foot solid wood equivalent for the residue. These residue returns, which could be negative if the price is a disposal cost, are added to the product revenue; the final total is returned to subroutine V. The macroequation described previously (equation 2) is used to perform the recovery calculations in subroutine YIELD.

Function CALIB—This function adjusts all the grade-yield fractions such that they sum to 1.00.

Function COST—The cost of the log, including delivery to the mill and product manufacture, is estimated by this function. As in subroutine YIELD, the macroequation (equation 2) is used to permit a wide variety of cost relations through selective zeroing of combinations of coefficients during the data entry with BUCKIT. The first action of function COST is to convert the log volume to cunits. Next, it sums six elements of woods-to-mill costs; stumpage, felling, bucking and limbing, yarding, loading, and hauling and unloading. Each of these is estimated in terms of dollars per cunit and multiplied by the log volume. An analyst may sometimes not wish to consider elements of delivered log cost in BUCKDP1 analyses; if so, simply zero all the coefficients for these six elements.

After the delivered-log cost is calculated, the appropriate product manufacturing cost in dollars per cunit of log is estimated, multiplied by the log volume, and added to previous costs.

The user must enter the cost data in BUCKDP1 in the same order that the markets were entered; that is, if the markets were entered in the order, PULP, LUMBER, and VENEER, then the cost coefficients must be in the same order. The TREEVAL system automatically ensures this coordination during data entry.

Subroutine RESULT—After the DP calculations for a tree are complete, subroutine RESULT identifies the optimal pattern of bucked logs, their end uses, and the optimal conversion return. This involves finding the optimal conversion return for the last stage in array F and scanning backward through array JP to identify the logs associated with the optimal solution. If JP contains a – 1, this means no logs should be cut at the current stage, and the program moves another stage down the stem. A positive integer in JP, however, refers to the position of a solution log in the list of log lengths. From the length of a solution log and its stage position in the tree, other log characteristics can be recalculated. The reason for recalculating solution logs is to save computer memory; storing all characteristics of the log that could be cut at each stage will greatly increase memory requirements. If desired (IOUT = 3), a printout is made of the individual logs comprising the pattern, starting with the uppermost log. If the output code includes stand statistics (IOUT <> 3), the log characteristics—multiplied by the number of trees in the tree class—are accumulated into tree-class and stand-statistics by subroutine SUM. Because the current solution log has its small end located at a specific stage, the number of stages comprising the log's length is deducted to indicate the stage location of the large end of the log. This stage at the large end is where RESULT continues the process of identifying solution logs.

When the solution scan is complete, the remainder of the output is printed according to the option chosen. This may be the total conversion return for the current tree if the detailed pattern option (IOUT = 3) was chosen. For each tree class, a summary can be printed that includes number of pieces, value, volume, and average diameter by product (IOUT = 2, 4) along with the overall averages for the entire stand. Alternatively, (IOUT = 1,4) the summary for each tree class may be output to data files. The companion users guide (Sachet and others 1989) describes these file options in greater detail.

Subroutine SUM—For the tree class (TABLE) and total stand (STAT), SUM accumulates for each product category the number of logs, total log volume, total conversion return, and log diameter by multiplying the resulting values for the representative tree by the number of trees in the class. The diameters and all total stand statistics are subsequently converted to averages by subroutine RESULT before being printed.

Subroutine HEAD—Subroutine HEAD writes a header whenever the summary table for the stand is printed.

Subroutine OUT—This subroutine prints the summary table for the current tree class.

BUCKSM1 : Simulation of Stem Bucking

Simulation Method

BUCKSM1 is very similar to BUCKDP1 in that most of the subroutines in BUCKDP1 are unchanged in BUCKSM1. The main difference is one of philosophy. Whereas BUCKDP1 sought to use market, price, cost and recovery data to find the highest value combination of logs that could be produced from the stem, BUCKSM1 reads an existing bucking pattern along with the tree data and calculates the value of the bucking solution provided by the user. The two programs give identical results when the optimal solution from BUCKDP1 is entered as data into BUCKSM1.

1. Read in the log lengths to be considered, arranged from shortest to longest
2. Read in corresponding marketing alternatives for each log length and the range of acceptable diameters in each market
3. Read in the product recovery data, prices for products and residues, and manufacturing costs
4. Read taper equation coefficients
5. For each tree (or representative of a tree class)
6. Read the tree description: d.b.h., total height, stump height, number of trees, number of logs
7. For each log in the bucking pattern
8. Read in its length and market designation
9. Calculate d.i.b. of both log ends by using taper equation
10. Calculate cubic foot volume of the log
11. Estimate product recovery and, if lumber or veneer, also estimate grade distribution and residues
12. Apply prices to estimates of products and residue volumes to get total revenue from the log
13. Apply costs to get conversion return
14. Next log
15. Print detailed solution for current tree (optional) and accumulate for stand summary
16. Next tree
17. Print stand summary (optional) or write summary to file (optional), or both
18. End

Figure 5—Computational overview of BUCKSIM1.

The purpose of BUCKSM1 is to provide a way to evaluate and compare current practices or simplified rules with what would occur if each tree was bucked by BUCKDP1. It uses the same assumptions described above for BUCKDP1. Figure 5 provides an overview of the procedures in BUCKSM 1.

BUCKSM1 Program Organization

Only those components differing from BUCKDP1 are described here for BUCKSM1. Most routines (INFO, LUMFAC, VENFAC, PLPFAC, DIAM, VOLUME, YIELD, CALIB, COST, SUM, HEAD, and OUT) are unchanged, and SCANER is unnecessary. Figure 6 presents the relations between the modules in BUCKSM1.

Main program BCKSM1—The main program BCKSM1 is nearly the same as its counterpart, BCKDP1. The principal exception is that the number of pieces (NBREAK) read from the tree file is not zero because this represents the number of logs in the bucking pattern. These logs must be listed from stump to top. The only other difference is that the call to subroutine DPRO is replaced by a call to subroutine SIM.

Subroutine SIM—SIM is a counterpart to subroutine DPRO and performs a similar role. It starts a loop that, for each log in the supplied pattern, reads the log length and corresponding market use. It calculates the top-height and bottom-height positions of the log in the tree and invokes subroutine V, which estimates the conversion return for this log when used in the given market. These values are accumulated for all logs in the pattern, and various descriptors for each log are stored.

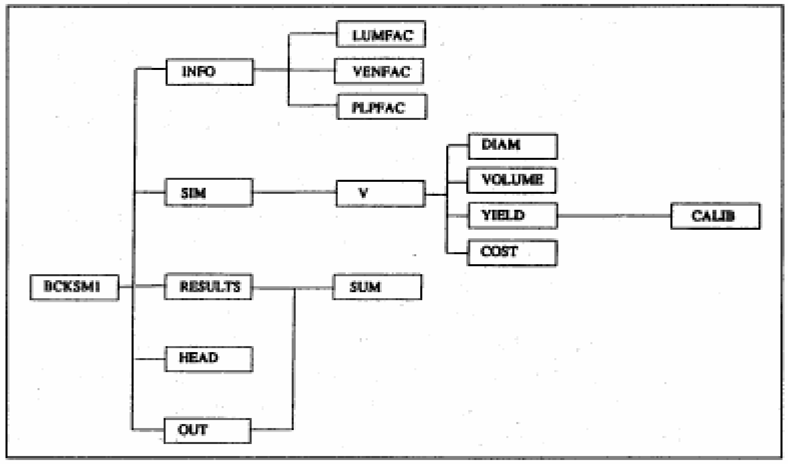


Figure 6—Program modules in BUCKSIM1.

Subroutine V—The only modification to subroutine V is elimination of the need to examine the log length over all markets. A check is done to see if the log length is among those listed as acceptable in the market, but a check is not done if its diameter is in range. If the length is not accepted by the designated market, the program will stop. This is assumed to occur only through a data-entry error. Except for these changes, this subroutine performs calculations for each product in the same way as described above.

Subroutine RESULT—Subroutine RESULT is simplified because it does not have to look up the solution from the values stored for DP stages. All it does is print the detailed pattern or summary table or write to data files in a fashion identical to that described for BUCKDP1.

Results and Discussion

Results from using BUCKDP1 and BUCKSM1 are illustrated in the users guide (Sachet and others 1989). The following sections present comments on using these programs and to explain techniques for situations that may seem unusual.

Use of BUCKDP1

Defining the DP-stage Interval— The user must define the stage interval (variable SEGL) as a number such that all the log lengths with trim included are an integer multiple of the DP-stage interval. If desired log lengths are 8.5, 9.0, 10.5, 12.5, 14.5, and 16.5 feet, then defining SEGL to be 0.5 (two stages per foot) would be a good choice. The log list would be the integers 17, 18, 21, 25, 29, and 33. Defining SEGL as 0.25 or 0.10 (4 or 10 stages per foot) will also work but will increase solution time by increasing the number of stages in the stem with no change in the final solution.

A difficulty can occur if, for example, the trim length is 4 inches or one-third of a foot. Entering SEGL as 0.33 (three stages per foot) as opposed to 0.333 may yield differ-- ent results. Such values should be entered to four or more places to prevent anomalies that may arise from rounding.

If some lengths have no trim allowance and others have a 4- or 6-inch trim, a log-length list might look like this: 8-1/3, 9.0,10-1/3, 12-1/3, 14-1/3, 16.5. The common divisor converting all these to integers is SEGL = 0.16667 (six stages per foot) yielding integers 50, 54, 62, 74, 86, and 99. Alternatively, a compromise (using, say three stages per foot), may be appropriate to reduce the number of stages by half and reduce solution time.

Defining pulp and chip logs and a scaling anomaly—In some analyses, the user may want to state that any portion of the. tree can be bucked for pulping or chipping. One easy way to do this is to define a log length equal to the stage interval for either or both of these markets. In other words, in integer terms, the log length is one. If this is done, BUCKDP1 may, because of the scaling methods used, produce a pattern with some apparent anomalies. Suppose, for example, the usable bole is 10 feet long and the log lengths are 0.5 foot for pulp, 8.5 feet for lumber, and 10.5 feet for lumber. Defining SEGL = 0.5, these convert to 1, 17, and 21 intervals, respectively. The solution would be expected to show the 8.5-foot saw log occupying the large end of the bole with 1.5 feet of pulp wood at the small end; however, because saw logs are scaled with rounded diameters and pulp logs are scaled with actual diameter, BUCKDP1 may find circumstances where it will get the same saw-log volume and conversion return no matter where the saw log is located in the 10-foot bole. Using actual diameters, BUCKDP1 will find more volume and conversion return if it takes the pulp from the large end rather than the small end. Thus, the optimal pattern may be the reverse of expectations and actually worth a small amount more. This could lead to solutions not intuitively pleasing, although the overall effect on stem value usually is quite small.

If a pulp or chip log is defined to be the same as SEGL, the detailed-pattern solution may list many contiguous pulp or chip lily pads, which reflects a long zone of the stem used by one of these markets. To overcome this, and to eliminate the possibility of solutions with unrealistically short pulp or chip logs, a specification can be made that pulp and chip logs have a range within which any length is acceptable; for example, pulp logs from 8 to 16 feet may be acceptable. If the SEGL based on other products is 0.5, then the list must include lengths of 8.0, 8.5, 9.0,..., 16.0, all of which are flagged for pulp and may also be flagged as acceptable in one or more of the other markets. Although this realistically eliminates the anomalies, it creates another problem: solution time may increase because of the expanded list of log lengths.

Another way to restrict pulp and chip logs is to limit the maximum diameter allowed for these products. The maximum diameter would normally be very large so that any portion of any tree could be evaluated for pulp or chips; in the presence of lumber and veneer markets, however, restricting pulp and chips to tops of large trees and to small trees might be appropriate. This can be done easily by limiting the maximum diameter accepted for these products. A 10 inch limit would eliminate pulp- and chip-log considerations until the d.i.b. of trees was below 10 inches, effectively limiting the analysis.

Use of BUCKSM1

If an optimal solution found with BUCKDP1 is submitted to BUCKSM1 , the latter may not always produce identical results, although this should be expected. Such discrepancies are due to rounding as the examples show.

1. Suppose BUCKDP1 presents a solution based on a stage interval of 0.33. The BUCKDP1 output formats may show a solution log as 17 feet and not as the 51 x 0.33 = 16.83 feet actually used to calculate volumes in BUCKDP1. The entry of 17 as data into BUCKSM1 introduces a volume and value difference.

2. BUCKDP1 may show 20 contiguous 0.5-foot stages used as pulp. The analyst converts this to 10 feet to enter into BUCKSM1. The cubic-volume formula may not yield the same result when they sum twenty 0.5-foot sections as opposed to calculating a single value based on 10 feet.

Computational Time

Considerable experience with BUCKSM1 and BUCKDP1 on both an IBM-PC and the University of Washington CYBER 180/855 mainframe computer indicates that solution times with BUCKSM1 are generally trivial.[8] Solution times with BUCKDP1 will take longer than BUCKSM1. The actual time will increase with:

- The number of trees or tree classes submitted for analysis.
- The size of trees.
- The market complexity.
- The number of log lengths in the log list.
- The number of markets using each length.
- Definition of a smaller DP-stage interval, which increases the number of stages in each tree.

Sensitivity Analysis

The TREEVAL package provides the analyst with many opportunities to perform sensitivity analyses. Any of the input data can be changed easily via menu selection. The simplest forms of sensitivity analyses would be to change the mix of markets, change the product prices, or change the log specifications. Examples are:

- What is the effect of deleting the veneer market?
- What is the effect of a 10-percent increase on the prices for veneer relative to lumber?
- What if the assumed sawmill processed logs from 10 to 16 feet and could process only those logs less than 16 inches in diameter?

TREEVAL allows systematic changes of one variable of interest at a time or a combination. TREEVAL also permits changes in the recovery relations and in manufacturing costs; the main limitation may be a lack of local information that is consistent with TREEVAL. Procedures in TREEVAL are based on cubic log scale and, for product-recovery relations, assume methods consistent with the recovery study procedures of the Timber Quality Research Project of the Pacific Northwest Research Station (see footnote 2). TREEVAL accepts any form of relation for lumber and veneer recovery consistent with equation (2); all the user must do is enter zeros for

---

[8] The use of trade, firm, or corporation names in this publication is for the information and convenience of the reader. Such use does not constitute an official endorsement or approval by the U.S. Department of Agriculture of any product or service to the exclusion of others that may be suitable.

inappropriate terms. If, for example, only an average recovery factor is available, then enter the average for constant a1 and zero the remaining coefficients during data entry. TREEVAL does not make any assumptions on product grade nomenclature. The user tells TREEVAL how many grades there are, gives an identifier for each, and fills out the appropriate terms in equation (2). As an example of this flexibility, we have used TREEVAL to evaluate red alder *(Alnus rubra* Bong.) as well as Douglas-fir.

The treatment of costs is similar to that used for recovery relations. Manufacturing cost is assumed to be a relation between log diameter and cost per cunit of log volume. TREEV AL also uses equation (2) as a "macro" for alternative cost expressions.

Even though the primary intent of TREEVAL was to aid in planning silvicultural regimes, it should be useful in other areas. One example is appraising an existing stand including the bucking strategies and log allocation. Another example is comparing a stand today with what it may become at some future time. The user must, of course, "project" the stand by whatever means seems appropriate.

Metric Equivalents

When you know:	Multiply by:	To find:
Inches	2.54	Centimeters
Feet	0.305	Meters
Square feet	0.093	Square meters
Cubic feet	0.028	Cubic meters
Acres	0.404	Hectares
Pounds	0.453	Kilograms

Literature Cited

Briggs, David G. 1977. A dynamic programming model for bucking tree stems into Logs. Tropical Forest Utilization System Rep. 5. Seattle, WA: College of Forest Resources, University of Washington. 12 p.

Briggs, David G. 1980. A dynamic programming approach to optimized stem conversion. Seattle WA: College of Forest Resources, University of Washington. 393 p. Ph.D. dissertation.

Bruce, David. 1982. Butt log volume estimators. Forest Science. 23(3): 489-503.

Bruce, David; Curtis, Robert O.; VanCoevering, Caryanne. 1968. Development of a system of taper and volume tables for red alder. Forest Science. 14(3): 339-350.

Deadman, M.W; Goulding, C.J. 1979. A method for assessment of recoverable volume by log types. New Zealand Journal of Forestry Science. 9(2): 225-239.

Faaland, Bruce; Briggs, David G. 1984. Log bucking and lumber manufacturing using dynamic programming. Management Science. 30(2): 245-257.

Fahey, Thomas D. 1987. Douglas-fir veneer: green veneer grades, layup loss, and green veneer and plywood recovery. Forest Products Journal. 37(11/12): 45-50.

Geerts, J.M.P. 1979. Optimal crosscutting of timber. Wageningen, The Netherlands: Department of Forest Technique and Forest Products, Agricultural University. 8 p.

Gluck, P; Koch, W. 1973. Die optimale Rohholzausforung. Centralblatt für des Gesampte Forstwesen. 90: 192-228.

Gupta, Indra. 1986. Use of crown length to define stem form: segmented taper equation. Seattle, WA: College of Forest Resources, University of Washington. 135 p. Ph.D. dissertation.

Kozak, A.; Munro, D.D.; Smith, J.H.G. 1969. Taper functions and their application in forest inventory. Forestry Chronicle. 45(4): 278-283.

Lembersky, Mark R; Chi, un H. 1984. Decision simulators speed implementation and improve operations. Interfaces. 14(4): 1-15.

Nasberg, Mikael. 1985. Mathematical programming models for optimal log bucking. Linkoping Studies in Science and Technology: Dissertation 132. Linkoping, Sweden: Department of Mathematics, Linkoping University. 200 p.

Pnevmaticos, S.M; Mann, S.H. 1972. Dynamic programming in tree bucking. Forest Products Journal. 22(2): 26-30.

Sachet, Janet K. Ayer; Briggs, David G.; Fight, Roger D. 1989. Tree value system: users guide. Gen. Tech. Rep. PNW-GTR-234. Portland, OR: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station. 45 p.

Briggs, David G. 1989. Tree value system: description and assumptions. Gen. Tech. Rep. PNW-GTR-239. Portland, OR: U.S. Department of Agriculture, Forest Service, Pacific Northwest Research Station. 24 p.

TREEVAL is a microcomputer model that calculates tree or stand values and volumes based on product prices, manufacturing costs, and predicted product recovery. It was designed as an aid in evaluating management regimes. TREEVAL calculates values in either of two ways, one based on optimized tree bucking using dynamic programming and one simulating the results of user-specified bucking.

Keywords: Tree value, tree volume, product recovery, dynamic programming, bucking, microcomputer.

Pacific Northwest Research Station
319 S.W. Pine St.
P.O. Box 3890
Portland, Oregon 97208

U.S. Department of Agriculture
Pacific Northwest .Research Station
319 S.W. Pine Street
P.O. Box 3890 .
Portland, Oregon 97208