SAS/OR® 13.2 User's Guide: Project Management

The NETDRAW Procedure

This document is an individual chapter from *SAS/OR® 13.2 User's Guide: Project Management*.

The correct bibliographic citation for the complete manual is as follows: SAS Institute Inc. 2014. *SAS/OR® 13.2 User's Guide: Project Management*. Cary, NC: SAS Institute Inc.

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

August 2014

Chapter 9
The NETDRAW Procedure

Contents

Overview: NETDRAW Procedure

The NETDRAW procedure draws a network diagram of the activities in a project. Boxes (or nodes) are used to represent the activities, and lines (or arcs) are used to show the precedence relationships among the activities. Though the description of the procedure is written using project management terminology, PROC NETDRAW can be used to draw any network such as an organizational chart or a software flow diagram. The only information required by the procedure for drawing such a diagram is the name of each activity in the project (or node in the network) and a list of all its immediate successor activities (or nodes connected to it by arcs). Note that project networks are acyclic. However, the procedure can also be used to draw cyclic networks by specifying explicitly the coordinates for the nodes or by requesting the procedure to break the cycles in an arbitrary fashion.

The ACTNET statement in the NETDRAW procedure is designed to draw activity networks that represent a project in Activity-On-Node (AON) format. All network information is contained in SAS data sets. The input data sets used by PROC NETDRAW and the output data set produced by the procedure are as follows:

- The Network input data set contains the precedence information, namely, the activity-successor information for all the nodes in the network. This data set can be an Activity data set that is used as input to the CPM procedure or a Schedule data set that is produced by the CPM procedure, or it can even be a Layout data set produced by the NETDRAW procedure. The minimum amount of information that is required by PROC NETDRAW is the activity-successor information that can be obtained from any one of the preceding three possible types of data sets. The additional information in the input data set can be used by the procedure to add detail to the nodes in the diagram, and, in the case of the Layout data set, the procedure can use the _X_ and _Y_ variables to lay out the nodes and arcs of the diagram.

- The Annotate input data set contains the graphics and text that are to be annotated on the network diagram. This data set is used by the procedure through the Annotate facility in SAS/GRAPH software.

- The Layout output data set produced by PROC NETDRAW contains all the information about the layout of the network. For each node in the network, the procedure saves the (x, y) coordinates; for each arc between each pair of nodes, the procedure saves the (x, y) coordinates of each turning point of the arc in a separate observation. Using these values, the procedure can draw the network diagram without recomputing node placement and arc routing.

Two issues arise in drawing and displaying a network diagram: the layout of the diagram and the format of the display. The layout of the diagram consists of placing the nodes of the network and routing the arcs of the network in an appropriate manner. The format of the display includes the size of the nodes, the distance between nodes, the color of the nodes and arcs, and the information that is placed within each node. Several options available in the ACTNET statement enable you to control the format of the display and the layout of the diagram; these options and their uses are explained in detail later in this chapter.

Following is a list of some of the key aspects of the procedure:

- The Network input data set specifies the activities (or nodes) in the network and their immediate successors. The amount of information displayed within each node can be controlled by the ID= option and by the use of default variables in the data set.
- The procedure uses the node-successor information to determine the placement of the nodes and the layout of the arcs connecting the nodes.
- By default, PROC NETDRAW uses the topological ordering of the activity network to determine the x coordinates of the nodes. In a time-based network diagram, the nodes can be ordered according to any numeric, SAS date, time, or datetime variable (the ALIGN= variable) in the input data set.
- The network does not have to represent a project. You can use PROC NETDRAW to draw any network. If the network has no cycles, then the procedure bases the node placement and arc routing on the precedence relationships. Alternately, you can specify explicitly the node positions or use the ALIGN= variable, and let the procedure determine the arc routing.
- To draw networks with cycles, use the BREAKCYCLE option. Alternately, you can use the ALIGN= option or specify the node positions so that the procedure needs only to determine the arc routing. See Example 9.12 for an illustration of a cyclic network.
- The ZONE= option enables you to divide the network into horizontal bands or zones. This is useful in grouping the activities of the project according to some appropriate classification.
- The TREE option instructs PROC NETDRAW to check if the network is indeed a tree, and, if so, to exploit the tree structure in the node layout. This feature is useful for drawing organizational charts, hierarchical charts, and work break-down structures.
- PROC NETDRAW gives you the option of displaying the network diagram in one of three modes: graphics, line-printer, or full-screen. The default mode is graphics mode, which enables you to produce charts of high resolution quality. Graphics mode requires SAS/GRAPH software. See the section "Graphics Options" on page 693 for more information about producing high-resolution quality network diagrams. You can also produce line-printer quality network diagrams by specifying the LINEPRINTER (LP) option in the PROC NETDRAW statement. In addition to sending the output to either a plotter or printer, you can view the network diagram at the terminal in full-screen mode by specifying the FULLSCREEN (FS) option in the PROC NETDRAW statement. See the section "Full-Screen Options" on page 692 for more information about viewing network diagrams in full-screen mode.
- The full-screen version of the procedure enables you to move the nodes around on the screen (subject to maintaining the precedence order of the activities) and thus change the layout of the network diagram.
- The graphics version of the procedure enables you to annotate the network diagram using the Annotate facility in SAS/GRAPH software.

- The positions of the nodes and arcs of the layout determined by PROC NETDRAW are saved in an output data set called the Layout data set. This data set can be used again as input to PROC NETDRAW; using such a data set saves some processing time because the procedure does not need to determine the node and arc placement.
- If necessary, the procedure draws the network across page boundaries. The number of pages that are used depends on the number of print positions that are available in the horizontal and vertical directions.
- In graphics mode, the COMPRESS and PCOMPRESS options enable you to produce the network on one page. You can also control the number of pages used to create the network diagram with the HPAGES= and VPAGES= options.
- In graphics mode, the ROTATE and ROTATETEXT options enable you to produce a top-down tree diagram.

Getting Started: NETDRAW Procedure

The first step in defining a project is to make a list of the activities in the project and determine the precedence constraints that need to be satisfied by these activities. It is useful at this stage to view a graphical representation of the project network. In order to draw the network, you specify the nodes of the network and the precedence relationships among them. Consider the software development project that is described in the "Getting Started" section of Chapter 4, "The CPM Procedure." The network data are in the SAS data set SOFTWARE, displayed in Figure 9.1.

Figure 9.1 Software Project

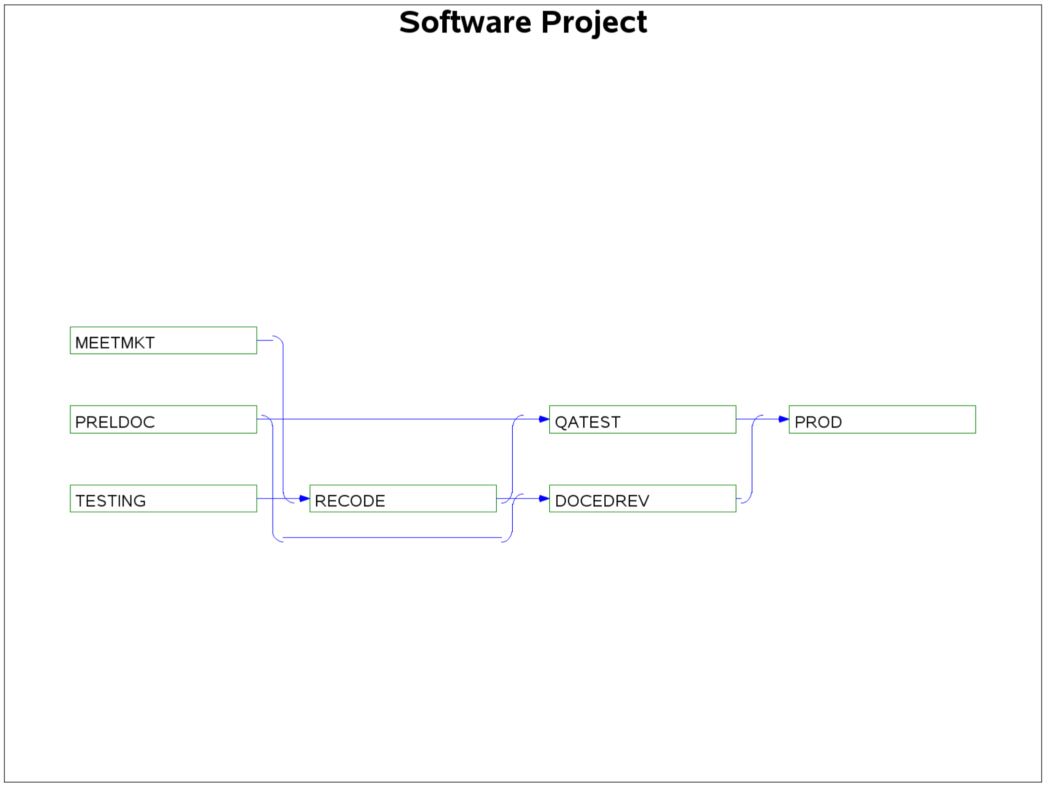
Software Project
Data Set SOFTWARE

Obs	descrpt	duration	activity	succesr1	succesr2
1	Initial Testing	20	TESTING	RECODE	
2	Prel. Documentation	15	PRELDOC	DOCEDREV	QATEST
3	Meet Marketing	1	MEETMKT	RECODE	
4	Recoding	5	RECODE	DOCEDREV	QATEST
5	QA Test Approve	10	QATEST	PROD	
6	Doc. Edit and Revise	10	DOCEDREV	PROD	
7	Production	1	PROD		

The following code produces the network diagram shown in Figure 9.2:

```
pattern1 v=e c=green;
pattern2 v=e c=red;

title h=3 'Software Project';
proc netdraw graphics data=software;
   actnet / act=activity htext=2
            succ=(succesr1 succesr2)
            pcompress separatearcs;
   run;
```

Figure 9.2 Software Project



The procedure determines the placement of the nodes and the routing of the arcs on the basis of the topological ordering of the nodes and attempts to produce a compact diagram. You can control the placement of the nodes by specifying explicitly the node positions. The data set SOFTNET, shown in Figure 9.3, includes the variables _X_ and _Y_, which specify the desired node coordinates. Note that the precedence information is conveyed using a single SUCCESSOR variable unlike the data set SOFTWARE, which contains two SUCCESSOR variables.

Figure 9.3 Software Project: Specify Node Positions

Software Project
Data Set SOFTNET

Obs	descrpt	duration	activity	succesor	_x_	_y_
1	Initial Testing	20	TESTING	RECODE	1	1
2	Meet Marketing	1	MEETMKT	RECODE	1	2
3	Prel. Documentation	15	PRELDOC	DOCEDREV	1	3
4	Prel. Documentation	15	PRELDOC	QATEST	1	3
5	Recoding	5	RECODE	DOCEDREV	2	2
6	Recoding	5	RECODE	QATEST	2	2
7	QA Test Approve	10	QATEST	PROD	3	3
8	Doc. Edit and Revise	10	DOCEDREV	PROD	3	1
9	Production	1	PROD		4	2

The following code produces a network diagram (shown in Figure 9.4) with the new node placement:

```
title h=3 'Software Project';
title2 h=2 'Controlled Layout';
proc netdraw graphics data=softnet;
   actnet / act=activity htext=1.25
            succ=(succesor)
            pcompress;
   run;
```

Figure 9.4 Software Project: Controlled Layout

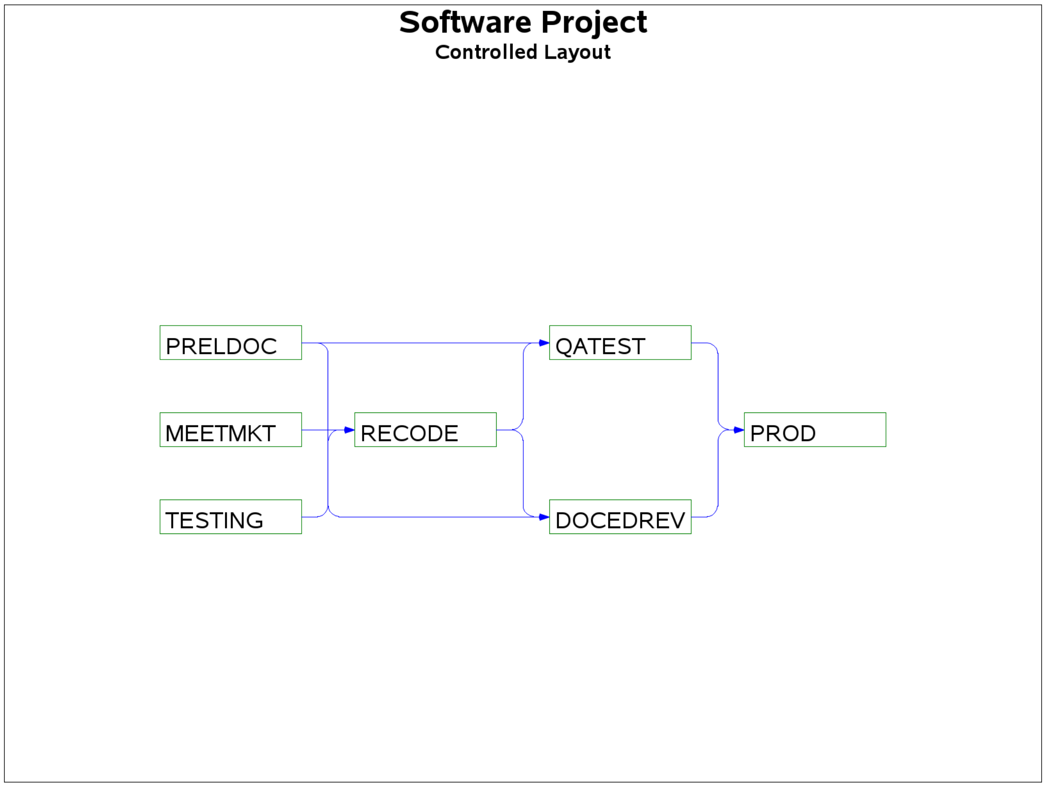


Figure 9.5 Software Project Schedule

Software Project
Project Schedule

descrpt	activity	succesr1	succesr2	duration	E_START
Initial Testing	TESTING	RECODE		20	01MAR04
Prel. Documentation	PRELDOC	DOCEDREV	QATEST	15	01MAR04
Meet Marketing	MEETMKT	RECODE		1	01MAR04
Recoding	RECODE	DOCEDREV	QATEST	5	21MAR04
QA Test Approve	QATEST	PROD		10	26MAR04
Doc. Edit and Revise	DOCEDREV	PROD		10	26MAR04
Production	PROD			1	05APR04

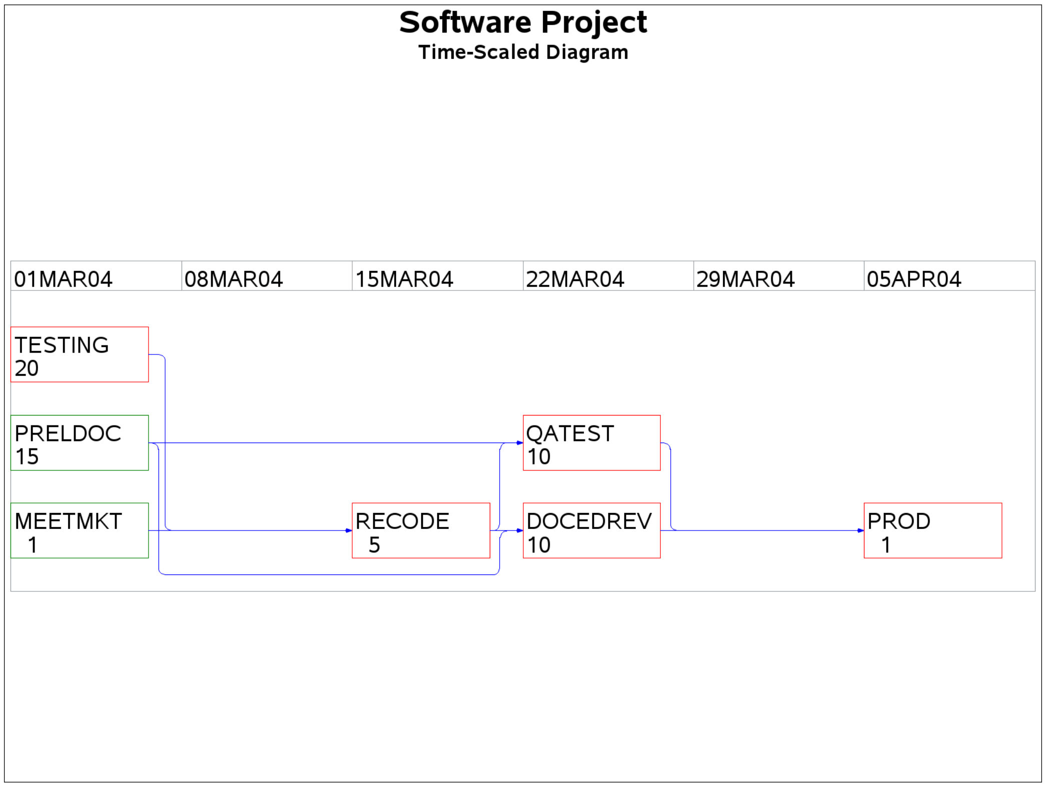
descrpt	E_FINISH	L_START	L_FINISH	T_FLOAT	F_FLOAT
Initial Testing	20MAR04	01MAR04	20MAR04	0	0
Prel. Documentation	15MAR04	11MAR04	25MAR04	10	10
Meet Marketing	01MAR04	20MAR04	20MAR04	19	19
Recoding	25MAR04	21MAR04	25MAR04	0	0
QA Test Approve	04APR04	26MAR04	04APR04	0	0
Doc. Edit and Revise	04APR04	26MAR04	04APR04	0	0
Production	05APR04	05APR04	05APR04	0	0

While the project is in progress, you may want to use the network diagram to show the current status of each activity as well as any other relevant information about each activity. PROC NETDRAW can also be used to produce a time-scaled network diagram using the schedule produced by PROC CPM. The schedule data for the software project described earlier are saved in a data set, INTRO1, which is shown in Figure 9.5.

To produce a time-scaled network diagram, use the TIMESCALE option in the ACTNET statement, as shown in the following program. The MININTERVAL= and the LINEAR options are used to control the time axis on the diagram. The ID=, NOLABEL, and NODEFID options control the amount of information displayed within each node. The resulting diagram is shown in Figure 9.6.

```
title h=3 'Software Project';
title2 h=2 'Time-Scaled Diagram';
proc netdraw graphics data=intro1;
   actnet / act=activity succ=(succ:)
            separatearcs pcompress htext=2
            timescale linear frame mininterval=week
            id=(activity duration) nolabel nodefid;
   run;
```

Figure 9.6 Software Project: Time-Scaled Network Diagram



Several other options are available to control the layout of the nodes, the appearance of the network, and the format of the time axis. For projects that have natural divisions, you can use the ZONE= option to divide the network into horizontal zones or bands. For networks that have an embedded tree structure, you can use the TREE option to draw the network like a tree laid out from left to right, with the root at the left edge of the diagram; in graphics mode, you can obtain a top-down tree with the root at the top of the diagram. For cyclic networks you can use the BREAKCYCLE option to enable the procedure to break cycles. All of these options are discussed in detail in the following sections.

Syntax: NETDRAW Procedure

The following statements are used in PROC NETDRAW:

PROC NETDRAW *options* ;
    ACTNET / *options* ;

Functional Summary

Table 9.1 outlines the options available for the NETDRAW procedure classified by function.

Table 9.1 Functional Summary

Description	Statement	Option
Color Options		
Specifies the color of arcs	ACTNET	CARCS=
Specifies the color of time axis	ACTNET	CAXIS=
Specifies the fill color for critical nodes	ACTNET	CCNODEFILL=
Specifies the color of critical arcs	ACTNET	CCRITARCS=
Specifies the outline color of critical nodes	ACTNET	CCRITOUT=
Specifies the fill color for nodes	ACTNET	CNODEFILL=
Specifies the outline color of nodes	ACTNET	COUTLINE=
Specifies the color of reference lines	ACTNET	CREF=
Specifies the color of reference break lines	ACTNET	CREFBRK=
Specifies the text color	ACTNET	CTEXT=
Data Set Specifications		
Specifies the Annotate data set	ACTNET	ANNOTATE=
Specifies the Annotate data set	NETDRAW	ANNOTATE=
Specifies the Activity data set	NETDRAW	DATA=
Specifies the Network output data set	NETDRAW	OUT=
Format Control Options		
Specifies the height of node in character cells	ACTNET	BOXHT=
Specifies the width of node in character cells	ACTNET	BOXWIDTH=
Specifies the DURATION variable	ACTNET	DURATION=
Specifies the ID variables	ACTNET	ID=
Suppresses default ID variables	ACTNET	NODEFID
Suppresses ID variable labels	ACTNET	NOLABEL
Specifies the upper limit on number of pages	ACTNET	PAGES=
Indicates completed or in-progress activities	ACTNET	SHOWSTATUS
Specifies the horizontal distance between nodes	ACTNET	XBETWEEN=
Specifies the vertical distance between nodes	ACTNET	YBETWEEN=
Full-Screen Options		
Specifies the reference break line character	ACTNET	BRKCHAR=
Specifies the characters for node outlines and connections	ACTNET	FORMCHAR=
Specifies the reference character	ACTNET	REFCHAR=
Graphics Catalog Options		
Specifies the description for the catalog entry	ACTNET	DESCRIPTION=
Specifies the name for the catalog entry	ACTNET	NAME=
Specifies the name of the graphics catalog	NETDRAW	GOUT=
Graphics Display Options		
Specifies the length of arrowhead in character cells	ACTNET	ARROWHEAD=
Centers each ID variable within node	ACTNET	CENTERID
Compresses the diagram to a single page	ACTNET	COMPRESS

Table 9.1 *continued*

Description	Statement	Option
Specifies the text font	ACTNET	FONT=
Specifies the text height	ACTNET	HEIGHT=
Specifies the horizontal margin in character cells	ACTNET	HMARGIN=
Specifies the number of horizontal pages	ACTNET	HPAGES=
Specifies the reference line style	ACTNET	LREF=
Specifies the reference break line style	ACTNET	LREFBRK=
Specifies the width of lines used for critical arcs	ACTNET	LWCRIT=
Specifies the width of lines	ACTNET	LWIDTH=
Specifies the width of outline for nodes	ACTNET	LWOUTLINE=
Suppresses filling of arrowheads	ACTNET	NOARROWFILL
Suppresses page number	ACTNET	NOPAGENUMBER
Suppresses vertical centering	ACTNET	NOVCENTER
Specifies the number of nodes in horizontal direction	ACTNET	NXNODES=
Specifies the number of nodes in vertical direction	ACTNET	NYNODES=
Displays page number at upper right corner	ACTNET	PAGENUMBER
Specifies the PATTERN variable	ACTNET	PATTERN=
Proportionally compresses the diagram	ACTNET	PCOMPRESS
Draws arcs with rectangular corners	ACTNET	RECTILINEAR
Reverses the order of the *y* pages	ACTNET	REVERSEY
Rotates the network diagram	ACTNET	ROTATE
Rotates text within node by 90 degrees	ACTNET	ROTATETEXT
Draws arcs along distinct tracks	ACTNET	SEPARATEARCS
Specifies the vertical margin in character cells	ACTNET	VMARGIN=
Specifies the number of vertical pages	ACTNET	VPAGES=
Layout Options		
Breaks cycles in cyclic networks	ACTNET	BREAKCYCLE
Uses dynamic programming algorithm to route arcs	ACTNET	DP
Specifies the number of horizontal tracks between nodes	ACTNET	HTRACKS=
Routes arcs along potential node positions	ACTNET	NODETRACK
Disables use of dynamic programming to route arcs	ACTNET	NONDP
Blocks track along potential node positions	ACTNET	NONODETRACK
Restricts scope of arc layout algorithm	ACTNET	RESTRICTSEARCH
Uses spanning tree layout	ACTNET	SPANNINGTREE
Draws network as a tree, if possible	ACTNET	TREE
Specifies the number of vertical tracks between nodes	ACTNET	VTRACKS=
Line-Printer Options		
Specifies the reference break line character	ACTNET	BRKCHAR=
Specifies the characters for node outlines and connections	ACTNET	FORMCHAR=
Specifies the reference character	ACTNET	REFCHAR=
Mode Options		
Invokes full-screen version	NETDRAW	FULLSCREEN

Table 9.1 *continued*

Description	Statement	Option
Invokes graphics version	NETDRAW	GRAPHICS
Invokes line-printer version	NETDRAW	LINEPRINTER
Suppresses display of diagram	NETDRAW	NODISPLAY
Network Specifications		
Specifies the ACTIVITY variable	ACTNET	ACTIVITY=
Specifies the LAG variables	ACTNET	LAG=
Specifies the SUCCESSOR variables	ACTNET	SUCCESSOR=
Time-Scale Options		
Specifies the ALIGN variable	ACTNET	ALIGN=
Draws reference lines at every level	ACTNET	AUTOREF
Draws a border around the network diagram and axes	ACTNET	FRAME
Draws all vertical levels	ACTNET	LINEAR
Specifies the maximum number of empty columns between tick marks	ACTNET	MAXNULLCOLUMN=
Specifies the smallest interval per level	ACTNET	MININTERVAL=
Specifies the number of levels per tick mark	ACTNET	NLEVELSPERCOLUMN=
Suppresses time axis on continuation pages	ACTNET	NOREPEATAXIS
Omits the time axis	ACTNET	NOTIMEAXIS
Stops procedure if align value is missing	ACTNET	QUITMISSINGALIGN
Draws zigzag reference line at breaks	ACTNET	REFBREAK
Shows all breaks in time axis	ACTNET	SHOWBREAK
Draws time-scaled diagram	ACTNET	TIMESCALE
Uses format of variable and not default	ACTNET	USEFORMAT
Tree Options		
Centers each node with respect to subtree	ACTNET	CENTERSUBTREE
Specifies the order of the children of each node	ACTNET	CHILDORDER=
Separates the sons of a node for symmetry	ACTNET	SEPARATESONS
Uses spanning tree layout	ACTNET	SPANNINGTREE
Draws network as a tree, if possible	ACTNET	TREE
Web Options		
Specifies the Image map output data set	NETDRAW	IMAGEMAP=
Specifies the HTML variable	ACTNET	WEB=
Zone Options		
Divides network into connected components	ACTNET	AUTOZONE
Suppresses zone labels	ACTNET	NOZONELABEL
Specifies the ZONE variable	ACTNET	ZONE=
Labels zones	ACTNET	ZONELABEL
Sets missing pattern values using zone	ACTNET	ZONEPAT
Leaves extra space between zones	ACTNET	ZONESPACE

PROC NETDRAW Statement

PROC NETDRAW *options* **;**

The following options can appear in the PROC NETDRAW statement.

ANNOTATE=*SAS-data-set*

specifies the input data set that contains the appropriate annotate variables for the purpose of adding text and graphics to the network diagram. The data set specified must be an Annotate data set. See the section "Using the Annotate Facility" on page 715 for further details about this option.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC NETDRAW for producing a network diagram. If DATA= is omitted, the most recently created SAS data set is used. This data set, also referred to as the Network data set, contains the network information (ACTIVITY and SUCCESSOR variables) and any ID variables that are to be displayed within the nodes. For details about this data set, see the section "Network Input Data Set" on page 699.

FULLSCREEN

FS

indicates that the network be drawn in full-screen mode. This enables you to view the network diagram produced by NETDRAW in different scales; you can also move nodes around the diagram to modify the layout.

GOUT=*graphics-catalog*

specifies the name of the graphics catalog used to save the output produced by PROC NETDRAW for later replay. This option is valid only if the GRAPHICS option is specified.

GRAPHICS

indicates that the network diagram produced be of high-resolution quality. If you specify the GRAPHICS option, but you do not have SAS/GRAPH software licensed at your site, the procedure stops and issues an error message. GRAPHICS is the default mode.

IMAGEMAP=*SAS-data-set*

names the SAS data set that receives a description of the areas of a graph and a link for each area. This information is for the construction of HTML image maps. You use a SAS DATA step to process the output file and generate your own HTML files. The graph areas correspond to the link information that comes from the WEB variable in the Network data set. This gives you complete control over the appearance and structure of your HTML pages.

LINEPRINTER

LP

produces a network diagram of line-printer quality.

NODISPLAY

requests the procedure not to display any output. The procedure still produces the Layout data set containing the details about the network layout. This option is useful to determine node placement and arc routing for a network that can be used at a later time to display the diagram.

OUT=*SAS-data-set*
specifies a name for the output data set produced by PROC NETDRAW. This data set, also referred to as the Layout data set, contains the node and arc placement information determined by PROC NETDRAW to draw the network. This data set contains all the information that was specified in the Network data set to define the project; in addition, it contains variables that specify the coordinates for the nodes and arcs of the network diagram. For details about the Layout data set, see the section "Layout Data Set" on page 706.

If the OUT= option is omitted, the procedure creates a data set and names it according to the DATA*n* convention.

ACTNET Statement

ACTNET */ options* **;**

The ACTNET statement draws the network diagram. You can specify several options in this statement to control the appearance of the network. All these options are described in the current section under appropriate headings: first, all options that are valid for all modes of the procedure are listed, followed by the options classified according to the mode (full-screen, graphics, or line-printer) of invocation of the procedure.

General Options

ACTIVITY=*variable*
specifies the variable in the Network data set that names the nodes in the network. If the data set contains a variable called _FROM_, this specification is ignored; otherwise, this option is required.

ALIGN=*variable*
specifies the variable in the Network data set containing the time values to be used for positioning each activity. This options triggers the TIMESCALE option that adds a time axis at the top of the network and aligns the nodes of the network according to the values of the ALIGN= variable. The minimum and maximum values of this variable are used to determine the time axis. The format of this variable is used to determine the default value of the MININTERVAL= option, which, in turn, determines the format of the time axis.

AUTOREF
draws reference lines at every tick mark. This option is valid only for time-scaled network diagrams.

AUTOZONE
enables automatic zoning (or dividing) of the network into connected components. This option is equivalent to defining an automatic zone variable that associates a tree number for each node. The tree number refers to a number assigned (by the procedure) to each distinct tree of a spanning tree of the network.

BREAKCYCLE
breaks cycles by reversing the back arcs of the network. The back arcs are determined by constructing an underlying spanning tree of the network. Once cycles are broken, the nodes of the network are laid out using a topological ordering of the new network formed from the original network by ignoring the back arcs. The back arcs are drawn after determining the network layout. Note that only the back arcs go from right to left.

BOXHT=*boxht*

specifies the height of the box (in character cell positions) used for denoting a node. If this option is not specified, the height of the box equals the number of lines required for displaying all of the ID variable values for any of the nodes. See the ROTATETEXT option (under "Graphics Options") for an exception.

BOXWIDTH=*boxwdth*

specifies the width of the box (in character cell positions) used for denoting a node. If this option is not specified, the width of the box equals the maximum number of columns required for displaying all of the ID variable values for any of the nodes. See the ROTATETEXT option (under "Graphics Options") for an exception.

CENTERSUBTREE

positions each node at the center of the subtree that originates from that node instead of placing it at the midpoint of its children (which is the default behavior). Note that the nodes are placed at integral positions along an imaginary grid, so the positioning may not be exactly at the center. This option is valid only in conjunction with the TREE option.

CHILDORDER=*order*

orders the children of each node when the network is laid out using either the TREE or the SPANNINGTREE option. The valid values for this option are TOPDOWN and BOTTOMUP for default orientation, and LEFTRGHT and RGHTLEFT for rotated networks (drawn with the ROTATETEXT option). The default is TOPDOWN.

DP

causes PROC NETDRAW to use a dynamic programming (DP) algorithm to route the arcs. This DP algorithm is memory and CPU-intensive and is not necessary for most applications.

DURATION=*variable*

specifies a variable that contains the duration of each activity in the network. This value is used only for displaying the durations of each activity within the node.

FRAME

encloses the drawing area with a border. This option is valid only for time-scaled or zoned network diagrams.

HTRACKS=*integer*

controls the number of arcs that are drawn horizontally through the space between two adjacent nodes. This option enables you to control the arc-routing algorithm. The default value is based on the maximum number of successors of any node.

ID=*(variables)*

specifies the variables in the Network data set that are displayed within each node. In addition to the ID variables, the procedure displays the ACTIVITY variable, the DURATION variable (if the DURATION= option was specified), and any of the following variables in the Network data set: E_START, E_FINISH, L_START, L_FINISH, S_START, S_FINISH, A_START, A_FINISH, T_FLOAT, and F_FLOAT. See Chapter 4, "The CPM Procedure," for a description of these variables. If you specify the NODEFID option, only the variables listed in the ID= option are displayed.

LAG=*variable*

LAG=*(variables)*

specifies the variables in the Network data set that identify the lag types of the precedence relationships between an activity and its successors. Each SUCCESSOR variable is matched with the corresponding LAG variable; that is, for a given observation, the *i*th LAG variable defines the relationship between the activities specified by the ACTIVITY variable and the *i*th SUCCESSOR variable. The LAG variables must be character type, and their values are expected to be specified as one of FS, SS, SF, or FF, which denote 'Finish-to-Start', 'Start-to-Start', 'Start-to-Finish', and 'Finish-to-Finish', respectively. You can also use the *keyword_duration_calendar* specification used by the CPM procedure, although PROC NETDRAW uses only the *keyword* information and ignores the lag *duration* and the lag *calendar*. If no LAG variables exist or if an unrecognized value is specified for a LAG variable, PROC NETDRAW interprets the lag as a 'Finish-to-Start' type.

This option enables the procedure to identify the different types of nonstandard precedence constraints (Start-to-Start, Start-to-Finish, and Finish-to-Finish) on graphics quality network diagrams by drawing the arcs from and to the appropriate edges of the nodes.

LINEAR

plots one column for every *mininterval* between the minimum and maximum values of the ALIGN= variable. By default, only those columns that contain at least one activity are displayed. This option is valid only for time-scaled network diagrams.

MAXNULLCOLUMN=*maxncol*

MAXEMPTY=*maxncol*

MAXZCOL=*maxncol*

MAXNCOL=*maxncol*

specifies the maximum number of empty columns between two consecutive nonempty columns. The default value for this option is 0. Note that specifying the LINEAR option is equivalent to specifying the MAXNULLCOLUMN= option to be infinity. This option is valid only for time-scaled network diagrams.

MININTERVAL=*mininterval*

specifies the smallest interval to be used per column of the network diagram. Thus, if MININTERVAL=DAY, each column is used to represent a day, and all activities that start on the same day are placed in the same column. The valid values for *mininterval* are SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, QTR, and YEAR. The default value of *mininterval* is determined by the format of the ALIGN= variable. The tick labels are formatted on the basis of *mininterval*; for example, if *mininterval* is DAY, the dates are marked using the DATE7. format, and if *mininterval* is HOUR, the labels are formatted as TIME5. and so on. This option is valid only for time-scaled network diagrams.

NLEVELSPERCOLUMN=*npercol*

NPERCOL=*npercol*

contracts the time axis by specifying that activities that differ in ALIGN= value by less than *npercol* units of MININTERVAL can be plotted in the same column. The default value of *npercol* is 1. This option is valid only for time-scaled network diagrams.

NODEFID

indicates that the procedure need not check for any of the default ID variables in the Network data set; if this option is in effect, only the variables specified in the ID= option are displayed within each node.

NODETRACK

specifies that the arcs can be routed along potential node positions if there is a clear horizontal track to the left of the successor (or _TO_) node. This is the default option. To prevent the use of potential node positions, use the NONODETRACK option.

NOLABEL

suppresses the labels. By default, the procedure uses the first three letters of the variable name to label all the variables that are displayed within each node of the network. The only exception is the variable that is identified by the ACTIVITY= option.

NONDP

uses a simple heuristic to connect the nodes. The default mode of routing is NONDP, unless the HTRACKS= or VTRACKS= option (or both) are specified and set to a number that is less than the maximum number of successors. The NONDP option is faster than the DP option.

NONODETRACK

blocks the horizontal track along potential node positions. This option may lead to more turns in some of the arcs. The default is NODETRACK.

NOREPEATAXIS

displays the time axis only on the top of the chart and not on every page. This option is useful if the different pages are to be glued together to form a complete diagram. This option is valid only for time-scaled network diagrams.

NOTIMEAXIS

suppresses the display of the time axis and its labels. Note that the nodes are still placed according to the time scale, but no axis is drawn. This option is valid only for time-scaled network diagrams.

NOZONELABEL

NOZONEDESCR

omits the zone labeling and the dividing lines. The network is still divided into zones based on the ZONE variable, but there is no demarcation or labeling corresponding to the zones.

PAGES=*npages*

specifies the maximum number of pages to be used for the network diagram in graphics and line-printer modes. The default value is 100.

QUITMISSINGALIGN

stops processing if the ALIGN= variable has any missing values. By default, the procedure tries to fill in missing values using the topological order of the network. This option is valid only for time-scaled network diagrams.

REFBREAK

shows breaks in the time axis by drawing a zigzag line down the diagram just before the tick mark at the break. This option is valid only for time-scaled network diagrams.

RESTRICTSEARCH

RSEARCH

restricts the scope of the arc layout algorithm by restricting the area of search for the arc layout when the DP option is in effect; this is useful in reducing the computational complexity of the dynamic programming algorithm. By default, using the DP algorithm to route the arcs, the *y* coordinates of the arcs can range through the entire height of the network. The RESTRICTSEARCH option limits the *y* coordinates to the minimum and the maximum of the *y* coordinates of the node and its immediate successors.

SEPARATESONS

separates the children (immediate successors) of a given node by adding an extra space in the center whenever it is needed to enable the node to be positioned at integral (x, y) coordinates. For example, if a node has two children, placing the parent node at the midpoint between the two children requires the *y* coordinate to be noninteger, which is not allowed in the Layout data set. By default, the procedure positions the node at the same *y* level as one of its children. The SEPARATESONS option separates the two children by adding a dummy child in between, thus enabling the parent node to be centered with respect to its children. This option is valid only in conjunction with the TREE option.

SHOWBREAK

shows breaks in the time axis by drawing a jagged break in the time axis line just before the tick mark corresponding to the break. This option is valid only for time-scaled network diagrams.

SHOWSTATUS

uses the variable STATUS (if it exists) in the Network data set to determine if an activity is in-progress or completed. Note that the STATUS variable exists in the Schedule data set produced by PROC CPM when used with an ACTUAL statement. If there is no STATUS variable or if the value is missing, the procedure uses the A_FINISH and A_START values to determine the status of the activity. If the network is drawn in line-printer or full-screen mode, activities in progress are outlined with the letter *P* and completed activities are outlined with the letter *F*; in high-resolution graphics mode, in-progress activities are marked with a diagonal line across the node from the bottom left to the top right corner, while completed activities are marked with two diagonal lines.

SPANNINGTREE

uses a spanning tree to place the nodes in the network. This method typically results in a wider layout than the default. However, for networks that have totally disjoint pieces, this option separates the network into connected components (or disjoint trees). This option is not valid for time-scaled or zoned network diagrams, because the node placement dictated by the spanning tree may not be consistent with the zone or the tickmark corresponding to the node.

SUCCESSOR=*(variables)*

specifies the variables in the Network data set that name all the immediate successors of the node specified by the ACTIVITY variable. This specification is ignored if the data set contains a variable named _TO_. At least one SUCCESSOR variable must be specified if the data set does not contain a variable called _TO_.

TIMESCALE

indicates that the network is to be drawn using a time axis for placing the nodes. This option can be used to align the network according to default variables. If the TIMESCALE option is specified without the ALIGN= option, the procedure looks for default variables in the following order: E_START, L_START, S_START, and A_START. The first of these variables that is found is used as the ALIGN= variable.

TREE

TREELAYOUT

requests the procedure to draw the network as a tree if the network is indeed a tree (that is, all the nodes have at most one immediate predecessor). The option is ignored if the network does not have a tree structure.

USEFORMAT

indicates that the explicit format of the ALIGN= variable is to be used instead of the default format based on the MININTERVAL= option. Thus, for example, if the ALIGN variable contains SAS date values, by default, the procedure uses the DATE7. format for the time axis labels irrespective of the format of the ALIGN= variable. The USEFORMAT option specifies that the variable's format should be used for the labels instead of the default format. This option is valid only for time-scaled network diagrams.

VTRACKS=*integer*

controls the number of arcs that are drawn vertically through the space between two adjacent nodes. A default value is based on the maximum number of successors of any node.

XBETWEEN=*integer*

HBETWEEN=*integer*

specifies the horizontal distance (in character cell positions) between two adjacent nodes. The value for this option must be at least 3; the default value is 5.

YBETWEEN=*integer*

VBETWEEN=*integer*

specifies the vertical distance (in character cell positions) between two adjacent nodes. The value for this option must be at least 3; the default value is 5.

ZONE=*variable*

names the variable in the Network data set used to separate the network diagram into zones.

ZONELABEL

ZONEDESCR

labels the different zones and draws dividing lines between two consecutive zones. This is the default behavior; to omit the labels and the dividing lines, use the NOZONELABEL option.

ZONESPACE

ZONELEVADD

draws the network with an extra row between two consecutive zones.

Full-Screen Options

BRKCHAR=*brkchar*

specifies the character used for drawing the zigzag break lines down the chart at break points of the time axis. The default value is >. This option is valid only for time-scaled network diagrams.

CARCS=*color*
specifies the color of the connecting lines (or arcs) between the nodes. The default value of this option is CYAN.

CAXIS=*color*
specifies the color of the time axis. The default value is WHITE. This option is valid only for time-scaled network diagrams.

CCRITARCS=*color*
specifies the color of arcs connecting critical activities. The procedure uses the values of the E_FINISH and L_FINISH variables (if they are present) in the Network data set to determine the critical activities. The default value is the value of the CARCS= option.

CREF=*color*
specifies the color of the reference lines. The default value is WHITE. This option is valid only for time-scaled network diagrams.

CREFBRK=*color*
specifies the color of the lines drawn to denote breaks in the time axis. The default value is WHITE. This option is valid only for time-scaled network diagrams.

FORMCHAR [*index list***]=***'string'*
specifies the characters used for node outlines and arcs. See the section "Line-Printer Options" on page 699 for a description of this option.

PATTERN=*variable*
specifies an integer-valued variable in the Network data set that identifies the color number for each node of the network. If the data set contains a variable called _PATTERN, this specification is ignored. All the colors available for the full-screen device are used in order corresponding to the number specified in the PATTERN variable; if the value of the PATTERN variable is more than the number of colors available for the device, the colors are repeated starting once again with the first color. If a PATTERN variable is not specified, the procedure uses the first color for noncritical activities, the second color for critical activities, and the third color for supercritical activities.

REFCHAR=*refchar*
specifies the reference character used for drawing reference lines. The default value is "|". This option is valid only for time-scaled network diagrams.

ZONEPAT
indicates that if a PATTERN variable is not specified or is missing and if a ZONE= variable is present, then the node colors are based on the value of the ZONE= variable.

Graphics Options

ANNOTATE=*SAS-data-set*
specifies the input data set that contains the appropriate annotate variables for the purpose of adding text and graphics to the network diagram. The data set specified must be an Annotate data set. See the section "Using the Annotate Facility" on page 715 for further details about this option.

ARROWHEAD=*integer*
specifies the length of the arrowhead in character cell positions. You can specify ARROWHEAD = 0 to suppress arrowheads altogether. The default value is 1.

CARCS=*color*
specifies the color to use for drawing the connecting lines between the nodes. The default color depends on the GOPTIONS statement and the GSTYLE system option; see the section "ODS Style Templates" on page 717 for more information.

CAXIS=*color*
specifies the color of the time axis. This option is valid only for time-scaled network diagrams. The default color depends on the GSTYLE system option and the value of the CTEXT= option; see the section "ODS Style Templates" on page 717 for more information.

CCNODEFILL=*color*
specifies the fill color for all critical nodes of the network diagram. If you specify this option, the procedure uses a solid fill pattern (with the color specified in this option) for all critical nodes, ignoring any fill pattern specified in the PATTERN statements; the PATTERN statements are used only to obtain the color of the outline for these nodes unless you specify the CCRITOUT= option. The default value for this option is the value of the CNODEFILL= option, if it is specified; otherwise, the procedure uses the PATTERN statements to determine the fill pattern and color.

CCRITARCS=*color*
specifies the color of arcs connecting critical activities. The procedure uses the values of the E_FINISH and L_FINISH variables (if they are present) in the Network data set to determine the critical activities. The default value of this option is the value of the CARCS= option.

CCRITOUT=*color*
specifies the outline color for critical nodes. The default value for this option is the value of the COUTLINE= option, if it is specified; otherwise, it is the same as the pattern color for the node.

CENTERID
centers the ID values placed within each node. By default, character valued ID variables are left justified and numeric ID variables are right justified within each node. This option centers the ID values within each node.

CNODEFILL=*color*
specifies the fill color for all nodes of the network diagram. If you specify this option, the procedure uses a solid fill pattern with the specified color, ignoring any fill pattern specified in the PATTERN statements; the PATTERN statements are used only to obtain the color of the outline for the nodes, unless you specify the COUTLINE= option.

COMPRESS
draws the network on one physical page. By default, the procedure draws the network across multiple pages if necessary, using a default scale that allots one character cell position for each letter within the nodes. Sometimes, to get a broad picture of the network and all its connections, you may want to view the entire network on one screen. If the COMPRESS option is specified, PROC NETDRAW determines the horizontal and vertical transformations needed so that the network is compressed to fit on one screen.

COUTLINE=*color*
: specifies an outline color for all nodes. By default, the procedure sets the outline color for each node to be the same as the fill pattern for the node. This option is useful when used in conjunction with a solid fill using a light color. Note that if an empty fill pattern is specified, then the COUTLINE= option will cause all nodes to appear the same.

CREF=*color*
: specifies the color of the reference lines. This option is valid only for time-scaled network diagrams. The default color depends on the GSTYLE system option and the value of the CTEXT= option; see the section "ODS Style Templates" on page 717 for more information.

CREFBRK=*color*
: specifies the color of the zigzag break lines. This option is valid only for time-scaled network diagrams. The default color depends on the GSTYLE system option and the value of the CTEXT= option; see the section "ODS Style Templates" on page 717 for more information.

CTEXT=*color*
CT=*color*
: specifies the color of all text on the network diagram including variable names or labels, values of ID variables, and so on. The default color depends on the GOPTIONS statement and the GSTYLE system option; see the section "ODS Style Templates" on page 717 for more information.

DESCRIPTION=*'string'*
DES=*'string'*
: specifies a descriptive string, up to 40 characters in length, that appears in the description field of the master menu in PROC GREPLAY. If the DESCRIPTION= option is omitted, the description field contains a description assigned by PROC NETDRAW.

FILLPAGE
: causes the diagram on each page to be magnified (if necessary) to fill up the page.

FONT=*font*
: specifies the font of the text. The default font depends on the GOPTIONS statement and the GSTYLE system option; see the section "ODS Style Templates" on page 717 for more information.

HEIGHT=*h*
HTEXT=*h*
: specifies that the height for all text in PROC NETDRAW (excluding the titles and footnotes) be *h* times the value of the global HTEXT= option, which is the default text height specified in the GOPTIONS statement of SAS/GRAPH. The value of *h* must be a positive real number; the default value is 1.0.

HMARGIN=*integer*
: specifies the width of a horizontal margin (in number of character cell positions) for the network in graphics mode. The default width is 1.

HPAGES=*h*
NXPAGES=*h*
: specifies that the network diagram is to be produced using *h* horizontal pages. However, it may not be possible to use *h* horizontal pages due to intrinsic constraints on the output.

For example, PROC NETDRAW requires that every horizontal page should contain at least one *x* level. Thus, the number of horizontal pages can never exceed the number of vertical levels in the network. The exact number of horizontal pages used by the network diagram is given in the _ORNETDR macro variable. See the section "Macro Variable _ORNETDR" on page 716 for further details.

The appearance of the diagram with respect to the HPAGES= option is also influenced by the presence of other related procedure options. The HPAGES= option performs the task of determining the number of vertical pages in the absence of the VPAGES= option. If the COMPRESS or PCOMPRESS option is specified in this scenario, the chart uses one vertical page (unless the HPAGES= and VPAGES= options are specified). If neither the COMPRESS nor PCOMPRESS option is specified, the number of vertical pages is computed in order to display as much of the chart as possible in a proportional manner.

LREF=*linestyle*

specifies the linestyle (1-46) of the reference lines. The default linestyle is 1, a solid line. See Figure 8.5 in Chapter 8, "The GANTT Procedure," for examples of the various line styles available. This option is valid only for time-scaled network diagrams.

LREFBRK=*linestyle*

specifies the linestyle (1-46) of the zigzag break lines. The default linestyle is 1, a solid line. See Figure 8.5 in Chapter 8, "The GANTT Procedure," for examples of the various line styles available. This option is valid only for time-scaled network diagrams.

LWCRIT=*integer*

specifies the line width for critical arcs and the node outlines for critical activities. If the LWCRIT= option is not specified, the procedure uses the value specified for the LWIDTH= option.

LWIDTH=*integer*

specifies the line width of the arcs and node outlines. The default line width is 1.

LWOUTLINE=*integer*

specifies the line width of the node outlines. The default line width for the node outline is equal to LWIDTH for noncritical nodes and LWCRIT for critical nodes.

NAME=*'string'*

specifies a string of up to eight characters that appears in the name field of the catalog entry for the graph. The default name is NETDRAW. If either the name specified or the default name duplicates an existing name in the catalog, then the procedure adds a number to the duplicate name to create a unique name, for example, NETDRAW2.

NOARROWFILL

draws arrowheads that are not filled. By default, the procedure uses filled arrowheads.

NOPAGENUMBER

NONUMBER

suppresses the page numbers that are displayed in the top right corner of each page of a multipage network diagram. Note that the pages are ordered from left to right, bottom to top (unless the REVERSEY option is specified).

NOVCENTER
draws the network diagram just below the titles without centering in the vertical direction.

NXNODES=*nx*
specifies the number of nodes that should be displayed horizontally across each page of the network diagram. This option determines the value of the HPAGES= option; this computed value of HPAGES overrides the specified value for the HPAGES= options.

NYNODES=*ny*
specifies the number of nodes that should be displayed vertically across each page of the network diagram. This option determines the value of the VPAGES= option; this computed value of VPAGES overrides the specified value for the VPAGES= options.

PAGENUMBER

PAGENUM
numbers the pages of the network diagram on the top right-hand corner of the page if the diagram exceeds one page. The numbering scheme is from left to right, bottom to top (unless the REVERSEY option is specified).

PATTERN=*variable*
specifies an integer-valued variable in the Network data set that identifies the pattern for filling each node of the network. If the data set contains a variable called _PATTERN, this specification is ignored. The patterns are assumed to have been specified using PATTERN statements. If a PATTERN variable is not specified, the procedure uses the first PATTERN statement for noncritical activities, the second PATTERN statement for critical activities, and the third PATTERN statement for supercritical activities.

PCOMPRESS
draws the network diagram on one physical page. As with the COMPRESS option, the procedure determines the horizontal and vertical transformation needed so that the network is compressed to fit on one screen. However, in this case, the transformations are such that the network diagram is proportionally compressed. See Example 9.4 for an illustration of this option.

If the HPAGES= and VPAGES= options are used to control the number of pages, each page of the network diagram is drawn while maintaining the original aspect ratio.

RECTILINEAR
draws arcs with rectangular corners. By default, the procedure uses rounded turning points and rounded arc merges in graphics mode.

REVERSEY
reverses the order in which the *y* pages are drawn. By default, the pages are ordered from bottom to top in the graphics mode. This option orders them from top to bottom.

ROTATE
rotates the network diagram to change the orientation of the network to be from top to bottom instead of from left to right. For example, you can use this option to draw a Bill of Materials diagram that is traditionally drawn from top to bottom with the Final Product drawn at the top of the tree. In addition to rotating the orientation of the network, use the ROTATETEXT option to rotate the text within each node. See Example 9.18 for an illustration of this option.

This option is similar to the global graphics option, ROTATE (GOPTIONS ROTATE). Note that if the global graphics option is used, titles and footnotes also need to be drawn with an angle specification: A=90. However, some device drivers ignore the global graphics option, ROTATE (for example, the SASGDDMX driver). Use the ROTATE option on the ACTNET statement for such device drivers.

ROTATETEXT

RTEXT

rotates the text within the nodes by 90 degrees. This option is useful when used in conjunction with the ROTATE option in the ACTNET statement (or the global graphics option ROTATE) to change the orientation of the network to be from top to bottom instead of from left to right. For example, you can use this option to draw an organizational chart that is traditionally drawn from top to bottom with the head of the organization at the top of the chart. If the ROTATETEXT option is specified, then the definitions of the BOXHT= and BOXWIDTH= options are reversed. See Example 9.18 for an illustration of this option.

SEPARATEARCS

separates the arcs to follow distinct tracks. By default, the procedure draws all segments of the arcs along a central track between the nodes, which may cause several arcs to be drawn on top of one another. If the SEPARATEARCS option is specified, the procedure may increase the values of the XBETWEEN= and YBETWEEN= options to accommodate the required number of lines between the nodes.

VMARGIN=*integer*

specifies the width of a vertical margin (in number of character cell positions) for the network. The default width is 1.

VPAGES=*v*

NYPAGES=*v*

specifies that the network diagram is to be produced using *v* vertical pages. This, however, may not be possible due to intrinsic constraints on the output. For example, PROC NETDRAW requires that every vertical page should contain at least one *y* level. Thus, the number of vertical pages can never exceed the number of horizontal levels in the network. The exact number of vertical pages used by the procedure is provided in the _ORNETDR macro variable. See the section "Macro Variable _ORNETDR" on page 716 for further details.

The appearance of the diagram with respect to the VPAGES= option is also influenced by the presence of other related procedure options. The VPAGES= option performs the task of determining the number of horizontal pages in the absence of the HPAGES= option (or the NXNODES= option). If the COMPRESS or PCOMPRESS option is specified (without the HPAGES= or NXNODES= options), the chart uses one horizontal page. If neither the COMPRESS nor PCOMPRESS option is specified, the number of horizontal pages is computed in order to display as much of the chart as possible in a proportional manner.

WEB=*variable*

HTML=*variable*

specifies the character variable in the Network data set that identifies an HTML page for each activity. The procedure generates an HTML image map using this information for each node in the network diagram.

ZONEPAT

indicates that if a PATTERN= variable is not specified or is missing and if a ZONE= variable is present, then the node patterns are based on the value of the ZONE= variable.

Line-Printer Options

BRKCHAR=*brkchar*

specifies the character used for drawing the zigzag break lines down the chart at break points of the time axis. The default value is >. This option is valid only for time-scaled network diagrams.

FORMCHAR [*index list***]=***'string'*

specifies the characters used for node outlines and arcs. The value is a string 20 characters long. The first 11 characters define the 2 bar characters, vertical and horizontal, and the 9 corner characters: upper-left, upper-middle, upper-right, middle-left, middle-middle (cross), middle-right, lower-left, lower-middle, and lower-right. These characters are used to outline each node and connect the arcs. The nineteenth character denotes a right arrow. The default value of the FORMCHAR= option is `|----|+|---+=|-/\<>*`. Any character or hexadecimal string can be substituted to customize the appearance of the diagram. Use an index list to specify which default form character each supplied character replaces, or replace the entire default string by specifying the full character replacement string without an index list. For example, change the four corners of each node and all turning points of the arcs to asterisks by specifying

```
   FORMCHAR(3 5 7 9 11)= '*****'
```

Specifying

```
   formchar='           ' (11 blanks)
```

produces a network diagram with no outlines for the nodes (as well as no arcs). For further details about the FORMCHAR= option see Chapter 7, "The DTREE Procedure," and Chapter 8, "The GANTT Procedure."

REFCHAR=*refchar*

specifies the reference character used for drawing reference lines. The default value is "|". This option is valid only for time-scaled network diagrams.

Details: NETDRAW Procedure

Network Input Data Set

The Network input data set contains the precedence information, namely the activity-successor information for all the nodes in the network. The minimum amount of information that is required by PROC NETDRAW is the activity-successor information for the network. Additional information in the input data set can be used by the procedure to add detail to the nodes in the diagram or control the layout of the network diagram.

Three types of data sets are typically used as the Network data set input to PROC NETDRAW. Which type of data set you use depends on the stage of the project:

- The Activity data set that is input to PROC CPM is the first type. In the initial stages of project definition, it may be useful to get a graphical representation of the project showing all the activity precedence constraints.
- The Schedule data set produced by PROC CPM (as the OUT= data set) is the second type. When a project is in progress, you may want to obtain a network diagram showing all the relevant start and finish dates for the activities in the project, in addition to the precedence constraints. You may also want to draw a time-scaled network diagram, with the activities arranged according to the start or finish times corresponding to any of the different schedules produced by PROC CPM.
- The Layout data set produced by PROC NETDRAW (as the OUT= data set) is the third type. Often, you may want to draw network diagrams of the project every week showing updated information (as the project progresses); if the network logic has not changed, it is not necessary to determine the placement of the nodes and the routing of the arcs every time. You can use the Layout data set produced by PROC NETDRAW that contains the node and arc positions, update the start and finish times of the activities or merge in additional information about each activity, and use the modified data set as the Network data set input to PROC NETDRAW. The new network diagram will have the same layout as the earlier diagram but will contain updated information about the schedule. Such a data set may also be useful if you want to modify the layout of the network by changing the positions of some of the nodes. See the section "Controlling the Layout" on page 707 for details about how the layout information is used by PROC NETDRAW. If the Layout data set is used, it contains the variables _FROM_ and _TO_; hence, it is not necessary to specify the ACTIVITY= and SUCCESSOR= options. See Example 9.13 and Example 9.14 for illustrations of the use of the Layout data set.

The minimum information required by PROC NETDRAW from the Network data set is the variable identifying each node in the network and the variable (or variables) identifying the immediate successors of each node. In addition, the procedure can use other optional variables in the data set to enhance the network diagram. The procedure uses the variables specified in the ID= option to label each node. The procedure also looks for default variable names in the Network data set that are added to the list of ID variables; the default variable names are E_START, E_FINISH, L_START, L_FINISH, S_START, S_FINISH, A_START, A_FINISH, T_FLOAT, and F_FLOAT. The format used for determining the location of these variables within each node is described in the section "Format of the Display" on page 704. See the section "Variables in the Network Data Set" on page 701 for a table of all the variables in the Network data set and their interpretations by PROC NETDRAW.

If the Network data set contains the variables _X_ and _Y_ identifying the *x* and *y* coordinates of each node and each turning point of each arc in the network, then this information is used by the procedure to draw the network. Otherwise, the precedence relationships among the activities are used to determine the layout of the network. It is possible to specify only the node positions and let the procedure determine the routing of all the arcs. However, partial information cannot be augmented by the procedure.

NOTE: If arc information is provided, the procedure assumes that it is complete and correct and uses it exactly as specified.

Variables in the Network Data Set

The NETDRAW procedure expects all the network information to be contained in the Network input data set named by the DATA= option. The network information is contained in the ACTIVITY and SUCCESSOR variables. In addition, the procedure uses default variable names in the Network data set for specific purposes. For example, the _X_ and _Y_ variables, if they are present in the Network data set, represent the coordinates of the nodes, the _SEQ_ variable indexes the turning points of each arc of the network, and so on.

In addition to the network precedence information, the Network data set may also contain other variables that can be used to change the default layout of the network. For example, the nodes of the network can be aligned in the horizontal direction using the ALIGN= specification, or they can be divided into horizontal bands (or zones) using a ZONE variable.

Table 9.2 lists all of the variables associated with the Network data set and their interpretations by the NETDRAW procedure. Note that all the variables are identified to the procedure in the ACTNET statement. Some of the variables use default names that are recognized by the procedure to denote specific information, as explained previously. The table indicates if the variable is default or needs to be identified in the ACTNET statement.

Table 9.2 Network Data Set and Associated Variables

Statement	Variable Name	Interpretation
ACTNET	ACTIVITY	Activity or node name
	ALIGN	Align variable for time-scaled network
	DURATION	Duration of activity
	ID	Additional variables to be displayed
	PATTERN	Pattern number
	SUCCESSOR	Immediate successor
	WEB	HTML page corresponding to activity
	ZONE	Zone variable for dividing network
Default Variable Names	A_FINISH	Default ID variable
	A_START	Default ID variable
	E_FINISH	Default ID variable
	E_START	Default ID variable
	F_FLOAT	Default ID variable
	L_FINISH	Default ID variable
	L_START	Default ID variable
	S_FINISH	Default ID variable
	S_START	Default ID variable
	T_FLOAT	Default ID variable
	FROM	Supersedes ACTIVITY= specification
	_PATTERN	Supersedes PATTERN= specification
	SEQ	Index of turning point in arc
	_TO_	Supersedes SUCCESSOR= specification
	_X_	*x* coordinate of node or arc turning point
	_Y_	*y* coordinate of node or arc turning point

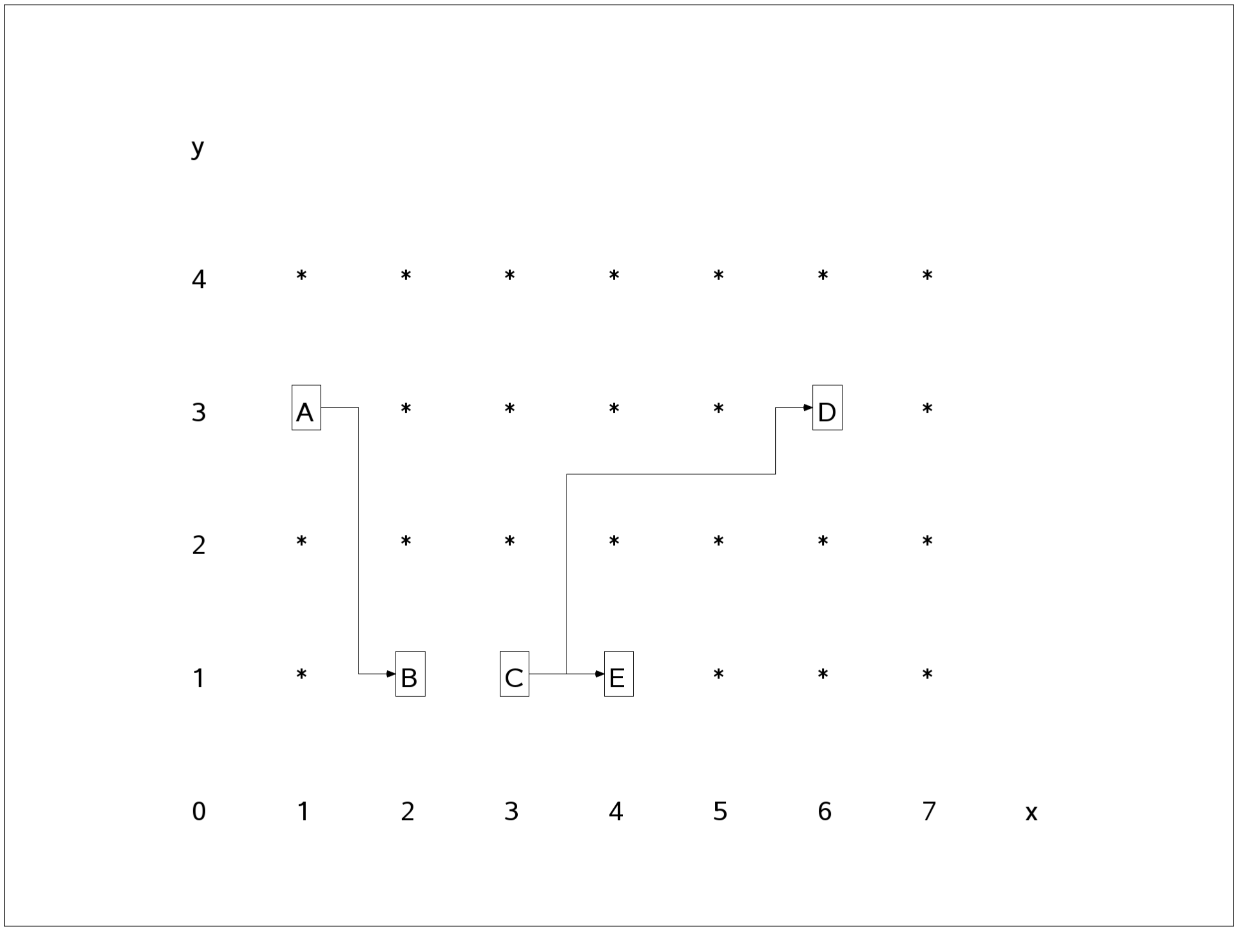
Missing Values

Missing values are not allowed for the ACTIVITY, _X_, _Y_, and _SEQ_ variables. Missing values for the SUCCESSOR and ID variables are ignored. Missing values are not allowed for the ALIGN= variable if the QUITMISSINGALIGN option is specified; otherwise, the procedure determines suitable values for the ALIGN= variable using the topological ordering of the network nodes.

Layout of the Network

The network layout is determined in two stages. First, the precedence relationships are used to determine the positions of the nodes, which are then used to determine a routing of the arcs. The positions of the nodes and arcs are identified by specifying their *x* and *y* coordinates in a grid. Figure 9.7 shows a sample grid and explains some of the conventions followed by PROC NETDRAW in determining the node and arc layout. This notation will be useful in later sections that describe the Layout data set and how you can control the layout of the diagram. The asterisks in the figure represent possible positions for the nodes of the network. The arcs are routed between the possible node positions. For example, node **A** has coordinates $(1, 3)$ and node **B** has coordinates $(2, 1)$. The arc connecting them has two turning points and is completely determined by the two pairs of coordinates $(1.5, 3)$ and $(1.5, 1)$; here, $x = 1.5$ implies that the position is midway between the *x* coordinates *1* and *2*.

Figure 9.7 Sample Grid and Coordinates for Node and Arc Layout



PROC NETDRAW sets $x = 1$ for all nodes with no predecessors; the x coordinates for the other nodes are determined so that each node is placed to the immediate right of all its predecessors; in other words, no node will appear to the left of any of its predecessors or to the right of any of its successors in the network diagram. The nodes are placed in topological order: a node is placed only after all its predecessors have been placed. Thus, the node-placement algorithm requires that there should be no cycles in the network. The y coordinates of the nodes are determined by the procedure using several heuristics designed to produce a reasonable compact diagram of the network. To draw a network that has cycles, use the BREAKCYCLE option, or you can specify the node coordinates or an ALIGN= variable to circumvent the requirement of a topological ordering of the nodes (see the second part of Example 9.12).

Note that the x and y coordinates fix only a relative positioning of the nodes and arcs. The actual distance between two nodes, the width and height of each node, and so on can be controlled by specifying desired values for the options that control the format of the display, namely, BOXHT=, BOXWIDTH=, and so on. See the section "Format of the Display" on page 704 for details about these options.

By default, the procedure routes the arcs using a simple heuristic that uses, at most, four turning points: the arc leaves the predecessor node from its right edge, turns up or down according to whether the successor is above or below the current node position, then tracks horizontally across to the vertical corridor just before the successor node, and then tracks in a vertical direction to meet the successor node. For example, see the tracking of the arc connecting nodes `C` and `D` in Figure 9.7.

For networks that include some nonstandard precedence constraints, the arcs may be drawn from and to the appropriate edges of the nodes, depending on the type of the constraint.

The default routing of the arcs may lead to an unbalanced diagram with too many arcs in one section and too few in another. The DP option in the ACTNET statement causes the procedure to use a dynamic programming algorithm to route the arcs. This algorithm tries to route the arcs between the nodes so that not too many arcs pass through any interval between two nodes. The procedure sets the maximum number of arcs that are allowed to be routed along any corridor to be equal to the maximum number of successors for any node. The HTRACKS= and VTRACKS=options enable you to set these maximum values: HTRACKS specifies the maximum number of arcs that are allowed to pass horizontally through any point while VTRACKS specifies the same for arcs in the vertical direction. See Example 9.7 for an illustration of the HTRACKS= option.

The layout of the network for time-scaled and zoned network diagrams is discussed in the section "Time-Scaled Network Diagrams" on page 708 and the section "Zoned Network Diagrams" on page 710, respectively. the section "Organizational Charts or Tree Diagrams" on page 710 describes the layout of the diagram when the TREE option is specified.

Format of the Display

As explained in the previous section, the layout of the network is determined by the procedure in terms of *x* and *y* coordinates on a grid as shown in Figure 9.7. The distance between nodes and the width and height of each node is determined by the values of the format control options: XBETWEEN=, YBETWEEN=, BOXHT=, and BOXWIDTH= . Note that if the ROTATETEXT option is specified (in graphics mode), then the definitions of the BOXHT= and BOXWIDTH= options are reversed.

The amount of information that is displayed within each node is determined by the variables specified by the ID= option, the number of default variables found in the Network data set, and whether the NOLABEL and NODEFID options are specified. The values of the variables specified by the ID= option are placed within each node on separate lines. If the NOLABEL option is in effect, only the values of the variables are written; otherwise, each value is preceded by the name of the ID variable truncated to three characters. Recall from the section "Syntax: NETDRAW Procedure" on page 682 that, in addition to the variables specified using the ID= option, the procedure also displays additional variables. These variables are displayed below the variables explicitly specified by the ID= option, in pre-determined relative positions within each node (see Table 9.3.)

Table 9.3 Display Format for the Variables within Each Node

ID1	
.	
.	
.	
ID*n*	
Activity *variable*	Duration *variable*
E_START	E_FINISH
L_START	L_FINISH
S_START	S_FINISH
A_START	A_FINISH
T_FLOAT	F_FLOAT

NOTE: If a node is identified as a successor (through a SUCCESSOR variable) and is never identified with the ACTIVITY variable, the ID values for this node are never defined in any observation; hence, this node will have missing values for all the ID variables.

If the SHOWSTATUS option is specified and the Network data set contains progress information (in either the STATUS variable or the A_START and A_FINISH variables), the procedure appropriately marks each node referring to activities that are completed or in progress. See Example 9.8 for an illustration of the SHOWSTATUS option.

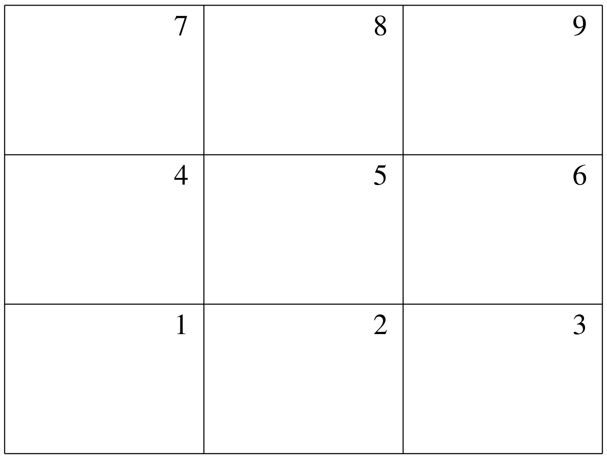
The features just described pertain to all three modes of the procedure. In addition, there are options to control the format of the display that are specific to the mode of invocation of the procedure. For graphics quality network diagrams, you can choose the color and pattern used for each node separately by specifying a different pattern number for the PATTERN variable, identified in the ACTNET statement (for details, see the section "Graphics Version" on page 714). For line-printer or full-screen network diagrams, the FORMCHAR= option enables you to specify special boxing characters that enhance the display; for full-screen network diagrams, you can also choose the color of the nodes using the PATTERN= option.

By default, all arcs are drawn along the center track between two consecutive nodes. The SEPARATEARCS option, which is available in the graphics version, separates arcs in the same corridor by drawing them along separate tracks, thus preventing them from being drawn on top of each other.

If the network fits on one page, it is centered on the page; in the graphics mode, you can use the NOVCENTER option to prevent centering in the vertical direction so that the network is drawn immediately below the title. If the network cannot fit on one page, it is split onto different pages appropriately. See the section "Page Format" on page 705 for a description of how the pages are split.

Page Format

Figure 9.8 Page Layout



As explained in the section "Format of the Display" on page 704, if the network fits on one page, it is centered on the page (unless the NOVCENTER option is specified); otherwise, it is split onto different pages appropriately, and each page is drawn starting at the bottom left corner. If the network is drawn on multiple pages, the procedure numbers each page of the diagram on the top right corner of the page. The pages are numbered starting with the bottom left corner of the entire picture. Thus, if the network diagram is broken into three horizontal and three vertical levels and you want to paste all the pieces together to form one picture, they should be arranged as shown in Figure 9.8.

The number of pages of graphical output produced by the NETDRAW procedure depends on several options such as the NXNODES=, NYNODES=, HPAGES=, VPAGES=, COMPRESS, PCOMPRESS, HEIGHT=, and the ID= options. The value of the HTEXT= option and the number of variables specified in the ID= options determines the size of each node in the network diagram, which in turn affects the number of horizontal and vertical pages needed to draw the entire network. The number of pages is also affected by the global specification of the HPOS=, VPOS=, HSIZE=, and VSIZE= graphics options.

The COMPRESS and PCOMPRESS options force the entire network diagram to be drawn on a single page. You can explicitly control the number of horizontal and vertical pages using the HPAGES= and VPAGES= options. The NXNODES= and NYNODES= options enable you to specify the number of nodes in the horizontal and vertical directions, respectively, on each page of the network diagram.

For examples of these options and how they affect the network diagram output, see Example 9.5.

Layout Data Set

The Layout data set produced by PROC NETDRAW contains all the information needed to redraw the network diagram for the given network data. In other words, the Layout data set contains the precedence information, the ID variables that are used in the current invocation of the procedure, and variables that contain the coordinate information for all the nodes and arcs in the network.

The precedence information used by the procedure is defined by two new variables named _FROM_ and _TO_, which replicate the ACTIVITY and SUCCESSOR variables from the Network data set. Note that the Layout data set has only one _TO_ variable even if the Network data set has multiple SUCCESSOR variables; if a given observation in the Network data set defines multiple successors for a given activity, the Layout data set defines a new observation for each of the successors. In fact, for each (node, successor) pair, a sequence of observations, defining the turning points of the arc, is saved in the Layout data set; the number of observations corresponding to each pair is equal to one plus the number of turns in the arc connecting the node to its successor. Suppose that a node ‘C’ has two successors, ‘D’ and ‘E,’ and the arcs connecting ‘C’ and ‘D’ and ‘C’ and ‘E’ are routed as shown in Figure 9.7. Then, Table 9.4 illustrates the format of the observations corresponding to the two (_FROM_, _TO_) pairs of nodes, (‘C’, ‘D’) and (‘C’, ‘E’).

Table 9.4 Sample Observations in the Layout Data Set

FROM	_TO_	_X_	_Y_	_SEQ_	_PATTERN	ID variables
C	D	3	1	0	1	
C	D	3.5	1	1	.	
C	D	3.5	2.5	2	.	
C	D	5.5	2.5	3	.	
C	D	5.5	3	4	.	
C	E	3	1	0	1	
		.			.	
		.			.	
		.			.	

For every (node, successor) pair, the first observation (_SEQ_ = ‘0’) gives the coordinates of the predecessor node; the succeeding observations contain the coordinates of the turning points of the arc connecting the predecessor node to the successor. The data set also contains a variable called _PATTERN, which contains the pattern number that is used for coloring the node identified by the _FROM_ variable. The value of this variable is missing for observations with _SEQ_ > 0.

Controlling the Layout

As explained in the section “Layout of the Network” on page 702, the procedure uses the precedence constraints between the activities to draw a reasonable diagram of the network. A very desirable feature in any procedure of this nature is the ability to change the default layout. PROC NETDRAW provides two ways of modifying the network diagram:

- using the full-screen interface
- using the Network data set

The full-screen method is useful for manipulating the layout of small networks, especially networks that fit on a handful of screens. You can use the full-screen mode to examine the default layout of the network and move the nodes to desired locations using the MOVE command from the command line or by using the appropriate function key. When a node is moved, the procedure reroutes all the arcs that connect to or from the node; other arcs are unchanged. For details about the MOVE command, see the section “Full-Screen Version” on page 711.

You can use the Network data set to modify or specify completely the layout of the network. This method is useful if you want to draw the network using information about the network layout that has been saved from an earlier invocation of the procedure. Sometimes you may want to specify only the positions of the node and let the procedure determine the routing of the arcs. The procedure looks for three default variables in the data set: _X_, _Y_, and _SEQ_ . The _X_ and _Y_ variables are assumed to denote the *x* and *y* coordinates of the nodes and all the turning points of the arcs connecting the nodes. The variable _SEQ_ is assumed to denote the order of the turning points. This interpretation is consistent with the values assigned to the _X_, _Y_, and _SEQ_ variables in the Layout data set produced by PROC NETDRAW. If there is no variable called _SEQ_ in the data set, the procedure assumes that only the node positions are specified and uses the specified coordinates to place the nodes and determines the routing of the arcs corresponding to these positions. If there is a variable called _SEQ_, the procedure requires that the turning points for each arc be specified in the proper order, with the variable _SEQ_ containing numbers sequentially starting with 1 and continuing onward. The procedure then draws the arcs exactly as specified, without checking for consistency or interpolating or extrapolating turning points that may be missing.

The ALIGN= variable provides another means of controlling the node layout (see the section “Time-Scaled Network Diagrams” on page 708). This variable can be used to specify the *x* coordinates for the different nodes of the network; the procedure then determines the *y* coordinates. Note that time-scaled network diagrams (without an ALIGN= specification) are equivalent to network diagrams drawn with the ALIGN= variable being set to the E_START variable.

You can also control the placement of the nodes using the ZONE=option (see the section “Zoned Network Diagrams” on page 710). The procedure uses the values of the ZONE variable to divide the network into horizontal zones. Thus, you can control the horizontal placement of the nodes using the ALIGN= option and the vertical placement of the nodes using the ZONE= option.

For networks that have a tree structure, the TREE option draws the network as a tree, thus providing another layout option (see the section "Organizational Charts or Tree Diagrams" on page 710). The procedure draws the tree from left to right, with the root at the left edge of the diagram. Thus, the children of each node are drawn to the right of the node. In the graphics mode of invocation, you can use the ROTATETEXT option in conjunction with the ROTATE option in the ACTNET statement (or the global graphics option ROTATE) to obtain a top-down tree diagram.

Time-Scaled Network Diagrams

By default, PROC NETDRAW uses the topological ordering of the activity network to determine the x coordinates of the nodes. As a project progresses, you may want to display the activities arranged according to their time of occurrence. Using the TIMESCALE option, you can draw the network with a time axis at the top and the nodes aligned according to their early start times, by default. You can use the ALIGN= option to specify any of the other start or finish times in the Network data set. In fact, PROC NETDRAW enables you to align the nodes according to any numeric variable in the data set.

If the TIMESCALE option is specified without any ALIGN= specification, the procedure chooses one of the following variables as the ALIGN= variable: E_START, L_START, S_START, or A_START, in that order. The first of these variables that is found is used to align the nodes. The minimum and maximum values of the ALIGN= variable are used to determine the time axis. The format of this variable is used to determine the default value for the MININTERVAL= option. The value of the MININTERVAL= option (or the default value) is used to determine the format of the time axis. You can override the format based on *mininterval* by specifying the desired format for the ALIGN= variable (using the FORMAT statement to indicate a standard SAS format or a special user-defined format) and the USEFORMAT option in the ACTNET statement. Table 9.5 lists the valid values of *mininterval* corresponding to the type of the ALIGN= variable and the default format corresponding to each value of *mininterval*. For each value in the first column, the first value of *mininterval* listed is the default value of the MININTERVAL= option corresponding to that type of the ALIGN= variable.

Several options are available in PROC NETDRAW to control the spacing of the nodes and the scaling of a time-scaled network diagram:

- The MININTERVAL= option enables you to scale the network diagram: one tick mark is associated with one unit of *mininterval*. Thus, if *mininterval* is DAY, each column is used to represent one day and all activities that start on the same day are placed in the same column. By default, the procedure omits any column (tick mark) that does not contain any node.
- The LINEAR option enables you to print a tick mark corresponding to every day (or the unit of *mininterval*). Note that, for a project that has few activities spread over a large period of time, the LINEAR option can lead to a network diagram that is very wide.
- The MAXNULLCOLUMN= option specifies the maximum number of empty columns that is allowed between two consecutive nonempty columns. The LINEAR option is equivalent to specifying *maxncol* = infinity, while the default time-scaled network diagram is drawn with *maxncol* = 0.

- The NLEVELSPERCOLUMN= option enables you to contract the network diagram by combining a few columns. For example, if *mininterval* is DAY and *nlevelspercol* is 7, each column contains activities that start within seven days of each other; note that the same effect can be achieved by setting *mininterval* to be WEEK.

Table 9.5 MININTERVAL Values and Axis Format

ALIGN Variable Type	MININTERVAL	Axis Label Format
number		numeric format
SAS time	HOUR	HHMM5.
	MINUTE	HHMM5.
	SECOND	TIME8.
SAS date	DAY	DATE7.
	WEEKDAY	DATE7.
	WEEK	DATE7.
	MONTH	MONYY5.
	QTR	MONYY5.
	YEAR	MONYY5.
SAS datetime	DTDAY	DATE7.
	WORKDAY	DATE7.
	DTWRKDAY	DATE7.
	DTSECOND	DATETIME16.
	DTMINUTE	DATETIME16.
	DTHOUR	DATETIME13.
	DTWEEK	DATE7.
	DTMONTH	MONYY5.
	DTQTR	MONYY5.
	DTYEAR	MONYY5.

The node-placement algorithm described in the section "Layout of the Network" on page 702 is modified slightly for time-scaled network diagrams. The *x* coordinate of each node is determined by the value of the ALIGN= variable. The scaling options just described are used to determine the tick mark corresponding to the node. The *y* coordinate is determined as before. Once the node placement is completed, the arc routing algorithm is the same as described earlier.

NOTE: Since the node placement for time-scaled networks is determined by the ALIGN= variable, it is possible that some of the arcs between the nodes may have to be routed from right to left instead of from left to right; in other words, there may be some backward arcs. Note also that, if the ALIGN= variable is used to determine the *x* coordinates of the nodes, the procedure can also draw networks that contain cycles (see the second part of Example 9.12).

Several other options are available to control the appearance of time-scaled network diagrams: AUTOREF, BRKCHAR=, CAXIS=, CREF=, CREFBRK=, FRAME, LREF=, LREFBRK=, NOREPEATAXIS, NOTIMEAXIS, REFBREAK, REFCHAR=, and SHOWBREAK. These options are described in the section "Syntax: NETDRAW Procedure" on page 682.

Zoned Network Diagrams

Most projects have at least one natural classification of the different activities in the project: department, type of work involved, location of the activity, and so on. The ZONE= option enables you to divide the network diagram into horizontal bands or zones corresponding to this classification. The procedure uses the following rules to place the nodes in a zoned network diagram:

- The values of the ZONE variable are used to define as many zones as there are distinct values of this variable.
- Each node of the network is drawn within its corresponding zone.
- The number of rows within each zone is determined by the maximum number of nodes in any given column that correspond to that zone.
- The values of the ZONE variable do not need to be sorted in any particular order, nor do they need to be grouped by distinct values.
- The zones are ordered according to the order of appearance of the different values of the ZONE variable in the Network data set. This enables you to choose any order for the zone values.
- For arcs that connect two nodes within the same zone, the arc lies entirely within the zone; in other words, all the turning points of the arc have *y* coordinates that are between the minimum and maximum *y* coordinates for the zone.
- Each zone is labeled by the value of the ZONE variable unless the NOZONELABEL option is specified.
- Each zone is separated from the next by a horizontal line drawn across the width of the network unless the NOZONELABEL option is specified.
- In the graphics and full-screen modes of invocation of the procedure, you can use the ZONEPAT option to color the nodes in each zone differently using different pattern statements. In the graphics mode, the first zone uses the first PATTERN statement, the second zone uses the second PATTERN statement, and so on; in full-screen mode, the colors available for the device are repeated in cyclic order. Note that the values of the PATTERN variable (or the default _PATTERN variable, if it exists in the Network data set) override the node patterns dictated by the ZONEPAT option.

Organizational Charts or Tree Diagrams

The NETDRAW procedure automatically draws any acyclic network; it does not have to be a representation of a project. You can also use the procedure to draw a general directed graph that has cycles, if node location is specified or if the BREAKCYCLE option is specified. The procedure attempts to draw the network in a compact fashion, which may not always produce the expected result. Trees form one such class of directed graphs that have an inherent natural layout that may not be produced by the default layout of PROC NETDRAW. The TREE option in the ACTNET statement exploits the tree structure of the network by laying the nodes out in the form of a tree.

A directed graph is said to be a tree if it has a root and there is a unique directed path from the root to every node in the tree. An equivalent characterization of a tree is that the root node has no predecessors and every other node has exactly one predecessor (Even 1979). Typical examples of trees that arise in project management are organizational charts or work breakdown structures. If the TREE option is specified, the NETDRAW procedure checks if the network has a tree structure and draws the network with the root at the left edge of the diagram and the children of each node appearing to the right of the node. In other words, the tree is drawn from left to right.

The NETDRAW procedure enables you to specify multiple trees in the same Network data set; each tree is drawn separately in the same diagram with all the roots appearing at the left edge of the diagram. Thus, you can use the TREE option as long as every node in the network has *at most* one predecessor. It you specify the TREE option and some node has multiple predecessors, the TREE option is ignored and the procedure uses the default node-layout algorithm.

There are several features that control the appearance of the tree:

- The children of each node are placed in the order of occurrence in the Network data set. The (x, y) coordinates of each node are required to be integers. The procedure attempts to place each node at the center of all its children, subject to the requirement that the coordinates must be integers. This requirement may cause some of the nodes to be positioned slightly off-center. See Example 9.15.
- The SEPARATESONS option separates the children of a node, if necessary, to enable the parent node to be exactly centered with respect to its children. See the second part of Example 9.15.
- The CENTERSUBTREE option can be used to center each node with respect to the entire subtree originating from the node instead of centering it with respect to its children.
- In graphics mode, you can change the orientation of the network to be from top to bottom instead of from left to right. To do so, use the ROTATETEXT option in the ACTNET statement to rotate the text within the nodes and the ROTATE option in the ACTNET statement (or the ROTATE global graphics option) to rotate the entire diagram by 90 degrees. See Example 9.18 for an illustration of this feature.

Full-Screen Version

You can invoke PROC NETDRAW in full-screen mode by specifying FS (or FULLSCREEN) in the PROC NETDRAW statement. The statement specifications are the same as for the line-printer mode. The full-screen mode offers you a convenient way to browse the network diagram of the project and change the layout of the network by moving the nodes of the network to desired locations. However, you cannot move a node to any position that violates the precedence constraints that must be satisfied by the node. In other words, you cannot move a node to the left of any of its predecessors or to the right of any of its successors. For time-scaled network diagrams, you cannot move a node out of the column corresponding to the value of the ALIGN= variable. For zoned network diagrams you cannot move a node out of its zone.

The format control options are treated in the same way as for the line-printer version, with some minor changes. It is assumed that the main purpose of invoking the procedure is to gain a general picture of the layout of the entire network and to modify it to some extent. In an effort to display as much of the network as possible, the initial display on the screen is drawn with only one row and three columns for each node. In other words, the BOXHT=, BOXWIDTH=, XBETWEEN=, and YBETWEEN= options are ignored by the procedure in drawing the initial display. However, the full-screen commands supported by PROC NETDRAW enable you to change the scale of the diagram. You can display as much or as little information within each node by invoking the SCALE ROW or the SCALE COL command or both. The SCALE MAX command causes the procedure to display the diagram using the values specified in the ACTNET statement or the dimensions that would be required to display all the ID information, whichever is larger. The SCALE RESET command returns the scaling to the initial values used for display.

The nodes of the network are color coded on the basis of the PATTERN variable. If there is no PATTERN variable, then the nodes are color coded depending on whether the activities are normal, critical, or supercritical. The nodes are drawn in reverse video. By default, the nodes are drawn without an outline; however, there is an OUTLINE command that lets you toggle back and forth between an outlined or non-outlined node. Using an outline for the node is useful if you want to obtain a printout of the screen display using SPRINT; it helps mark the boundary of each node clearly.

Commands

Table 9.6 lists the commands that can be invoked from the command line in the full-screen version of PROC NETDRAW. These commands are explained in greater detail in this section.

Table 9.6 Full-Screen Commands and Their Purposes

Scrolling	**Controlling Display**	**Changing Network Layout**	**Exiting**
BACKWARD	OUTLINE	CLEAR	GEND
FORWARD	SCALE	MOVE	END
LEFT			CANCEL
RIGHT			
TOP			
BOTTOM			
VSCROLL			
HSCROLL			

BACKWARD

scrolls toward the top of the network by the VSCROLL amount. BACKWARD MAX scrolls to the top of the network. You can specify the vertical scroll amount for the current command as BACKWARD PAGE | HALF | *n*.

BOTTOM

scrolls to the bottom of the network.

CANCEL

ends the current invocation of the procedure.

CLEAR

clears any outstanding move commands.

GEND

ends the current invocation of the procedure after drawing the network in graphics mode with the compress option.

END

ends the current invocation of the procedure.

FORWARD

scrolls toward the bottom of the network by the VSCROLL amount. FORWARD MAX scrolls to the bottom of the network. You can also specify the vertical scroll amount for the current command as FORWARD PAGE | HALF | *n*.

HELP

displays a help screen listing all the full-screen commands specific to PROC NETDRAW.

HOME

moves the cursor to the command line.

HSCROLL

sets the amount that information scrolls horizontally when you execute the LEFT or RIGHT command. The format is HSCROLL PAGE | HALF | *n*. The specification is assumed to be in number of horizontal levels. HSCROLL PAGE sets the scroll amount to be the number of horizontal levels that fit on one screen; HSCROLL HALF is half that amount; HSCROLL *n* sets the horizontal scroll amount to *n* levels.

KEYS

displays current function key settings for the NETDRAW procedure.

LEFT

scrolls toward the left boundary of the network by the HSCROLL amount. LEFT MAX scrolls to the left boundary. You can specify the horizontal scroll amount for the current command as LEFT PAGE | HALF | *n*.

MOVE

specifies a node to be moved or a place to move a node to. You can specify these in any order. Thus, you can first position the cursor on the node that you want to move, issue the MOVE command, and then position the cursor at a target position and issue the MOVE command again. If the target position is valid, the node is moved. You can also first specify the target position and then indicate the node that is to be moved.

NOTE: For a standard network, a node cannot be moved to any position that violates the topological ordering of the nodes in the network. For time-scaled network diagrams, you cannot move a node to a level corresponding to a different tick mark. For zoned network diagrams, you cannot move a node out of its zone.

OUTLINE

causes an outline to be drawn around each node in the network. This is useful if you want to print a copy of the screen by using the SPRINT command. The OUTLINE command works like an on/off switch: you can turn it off by entering the command again.

RIGHT

scrolls toward the right boundary of the network by the HSCROLL amount. RIGHT MAX scrolls to the right boundary. You can also specify the horizontal scroll amount for the current command as RIGHT PAGE | HALF | *n*.

SCALE

controls the scaling of the nodes and the space between nodes. The format of this command is SCALE MAX | MIN | RESET | ROW MAX | COL MAX | ROW MIN | COL MIN | ROW *n* | COL *n* | +*n* | -*n*. The number *n* denotes the number of character positions. SCALE MIN displays as many nodes on the screen as can fit. SCALE MAX enables as many rows and columns per node as is required to display all the information that pertains to it. SCALE ROW MAX displays the maximum number of rows per node. SCALE COL MAX displays the maximum number of columns per node. SCALE ROW *n* sets the number of rows per node to *n*. SCALE ROW $+n$ increases the number of rows per node by *n*. SCALE COL *n* sets the number of columns per node to *n*. SCALE COL $+n$ increases the number of columns per node by *n*. SCALE RESET sets the values to be the same as for the initial display. Note that none of these values can be greater than the dimensions of the screen.

TOP

scrolls to the top of the network.

VSCROLL

sets the amount by which information scrolls vertically when you execute the BACKWARD or FORWARD command. The format is VSCROLL PAGE | HALF | *n*. The specification is assumed to be in number of vertical levels. VSCROLL PAGE sets the scroll amount to be the number of vertical levels that fit on one screen; VSCROLL HALF is half that amount; VSCROLL *n* sets the vertical scroll amount to *n* levels.

Full-Screen Global Commands

Most of the global commands used in SAS/FSP software are also valid with PROC NETDRAW. Some of the commands used for printing screens are described in the section “Global Commands” on page 544 in Chapter 8, “The GANTT Procedure.”

Graphics Version

Several options are available in the ACTNET statement to enhance the appearance of the network diagram in graphics mode. These are described in the section “Graphics Options” on page 693. The format control options BOXWIDTH=, BOXHT=, XBETWEEN=, and YBETWEEN= are also valid in this mode and can be used to control the width and height of each node and the distance between the nodes. These parameters are specified in terms of number of character cell positions. The number of positions available on one page depends on the graphics device that is used; thus, if a plotter is used with large paper, more of the network will be drawn on a single page. Further, you can control the number of character cell positions on a page by changing the values of the global graphics options (HPOS= and VPOS=). Note that the NETDRAW procedure is not supported with the ActiveX or Java series of devices on the GOPTIONS statement.

You can also control the number of nodes on a given page by specifying the NXNODES= and NYNODES= options. The HPAGES= and VPAGES= options control the number of pages in the horizontal and vertical directions. Thus, you have a wide degree of control over the amount of information displayed on each page of the network diagram.

Another option that is available in graphics mode to control the appearance of your network diagrams is the specification of a PATTERN variable in the ACTNET statement. If the variable is named _PATTERN, you do not need to use the PATTERN= option; the procedure looks for such a variable by default. You can use this variable to specify the PATTERN definition that is to be used for filling each node of the network. Note that if the value of the _PATTERN variable is *j* for a particular node, PROC NETDRAW uses the specifications in the *jth generated* PATTERN *definition*, not the specifications in the PATTERN*j statement*.

The patterns that can be used with PROC NETDRAW are any of the patterns that can be used for drawing bars (not ones that are used for drawing maps). However, for the text to be visible, you may want to restrict the patterns used to be empty and change only the color of the pattern. You can also use solid fills with a light color and specify the COUTLINE= and CCRITOUT= options to mark noncritical and critical nodes with different colors for the outline.

See *SAS/GRAPH Software: Reference* for details about creating, canceling, reviewing, and altering PATTERN definitions. For a brief description of the PATTERN statement and for a list of available patterns, see Chapter 8, "The GANTT Procedure."

If a PATTERN variable is not specified, the procedure uses the values of the E_FINISH and L_FINISH variables (if these variables exist in the Network data set) to determine if activities in the project are normal, critical, or supercritical. The procedure then uses the first *generated* PATTERN *definition* to fill the nodes corresponding to noncritical activities, the second *generated* PATTERN *definition* for nodes corresponding to critical activities, and the third *generated* PATTERN *definition* for nodes corresponding to supercritical activities.

For zoned network diagrams, if there is no PATTERN variable, the ZONEPAT option enables you to color the nodes based on the values of the ZONE= variable.

Using the Annotate Facility

The Annotate facility enables you to enhance graphics output produced by PROC NETDRAW. To use this facility, you must create an Annotate data set, which contains a set of graphics commands that can be superimposed on the network diagram. This data set has a specific format and must contain key variables as described in *SAS/GRAPH Software: Reference*. The chapter entitled "The Annotate Data Set" lists the variables that are required in this data set and explains the coordinate systems used by the Annotate facility. The present section explains the use of data coordinates specifically with reference to the NETDRAW procedure.

When annotating a graph produced by any of the graphics procedures, it is helpful to use data coordinates that refer to the data values corresponding to the graph that is being annotated. For example, if you want to label a particular node of a network diagram with additional text, you can position the text accurately if you use data coordinates instead of screen coordinates. With respect to PROC NETDRAW, the Annotate facility uses the _X_ and _Y_ values in the Layout data set as the basis for the data coordinate system. To use this feature, you can invoke PROC NETDRAW (with the NODISPLAY option, if necessary) for the given network to produce the Layout data set that contains the _X_ and _Y_ coordinates for each node of the network. This data set can then be used to create the required Annotate data set containing the graphics commands positioning the primitives appropriately on the diagram using the data coordinates. See Example 9.16 and Example 9.17 for illustrations of this feature.

NOTE: The data coordinate system enables you to annotate the graph even if it spans multiple pages. However, each annotation must be entirely contained within a given page. For example, you cannot annotate a line on the network diagram that runs from one page of the diagram to another.

Web-Enabled Network Diagrams

The WEB variable enables you to define a HTML reference for each activity. This HTML reference is associated with the node corresponding to the activity. The WEB variable is a character variable and the values need to be of the form “HREF=htmlpage”.

In addition, you can also store the coordinate and link information defined by the WEB= option in a SAS data set by specifying the IMAGEMAP= option in the PROC NETDRAW statement. By processing this SAS data set using a DATA step, you can generate customized HTML pages for your network diagram.

Macro Variable _ORNETDR

The NETDRAW procedure defines a macro variable named _ORNETDR. This variable contains a character string that indicates the status of the procedure. It is set at procedure termination. The form of the _ORNETDR character string is STATUS= REASON= , where STATUS= is either SUCCESSFUL or ERROR_EXIT and REASON= (if PROC NETDRAW terminated unsuccessfully) can be one of the following:

CYCLE

BADDATA_ERROR

MEMORY_ERROR

IO_ERROR

SEMANTIC_ERROR

SYNTAX_ERROR

NETDRAW_BUG

UNKNOWN_ERROR

This information can be used when PROC NETDRAW is one step in a larger program that needs to determine whether the procedure terminated successfully or not. Because _ORNETDR is a standard SAS macro variable, it can be used in the ways that all macro variables can be used.

In addition to providing the “STATUS= REASON= ” string that indicates the status of the procedure, the macro variable _ORNETDR also provides some information about the network diagram produced by the current invocation of PROC NETDRAW.

The information given in _ORNETDR is described in the following list, along with the keyword that identifies it. These values refer to those actually used in producing the network diagram and are not necessarily the same as those specified in the invocation of the procedure.

- HPAGES= The number of horizontal pages
- VPAGES= The number of vertical pages
- SEGNAME= The name of the first network diagram segment in graphics mode

NOTE: Some of the information might be redundant or predictable in certain display modes. For example, the value of the SEGNAME= option is empty in line-printer and full-screen modes. The values of the HPAGES= and VPAGES= options are equal to 1 in full-screen mode.

Computer Resource Requirements

There is no inherent limit on the size of the network that you can draw with the NETDRAW procedure. Naturally, a sufficient amount of core memory must be available in order to invoke and initialize the SAS system. Furthermore, the amount of memory that is required depends on the mode of invocation of the procedure. The procedure attempts to store all the data in core memory. However, if the problem is too large to fit in core memory, the procedure resorts to using utility data sets and swaps between core memory and utility data sets as necessary.

The storage requirement for the data area that the procedure requires is proportional to the number of nodes and arcs in the network. You can further increase the memory that is required by specifying the DP option in the ACTNET statement. Recall that the DP option requests the use of a dynamic programming algorithm to route the arcs between the nodes, and such algorithms tend to grow exponentially with the size of the problem being solved.

ODS Style Templates

ODS style templates, or *styles*, control the overall look of your output. An ODS style template consists of a set of *style elements*. A style element is a collection of *style attributes* that apply to a particular feature or aspect of the output. You can specify a value for each attribute in a style. See Chapter 21, “Statistical Graphics Using ODS” (*SAS/STAT User’s Guide*), for a thorough discussion of ODS Graphics.

To create your own style or to modify a style for use with ODS Graphics, you need to understand the relationships between style elements and graph features. This information is provided in the ODS Graphics documentation at **`http://support.sas.com/documentation/onlinedoc/base/`**. You can create and modify style templates with the TEMPLATE procedure. For more information, see the section “TEMPLATE Procedure: Creating a Style Template” in the *SAS Output Delivery System: User’s Guide*. Kuhfeld (2010) also offers detailed information and examples.

PROC NETDRAW Style Template

A predefined ODS style template named NETDRAW is available for the NETDRAW procedure. You can use the template to maintain a consistent appearance in all graphical output produced by the procedure.

To change the current style, specify the STYLE= option in an ODS destination statement. The specified style is applied to all output for that destination until you change or close the destination or start a new SAS session. For example, the following statement specifies that ODS should apply the NETDRAW style template to all HTML output:

```
ods html style=netdraw;
```

To disable the use of graphical styles, specify the SAS system option NOGSTYLE.

The parent style template for the NETDRAW style is the DEFAULT style. Table 9.7 lists the style elements (in bold) and corresponding attributes specified in the NETDRAW style. The table also indicates which, if any, PROC NETDRAW options or graphics options (in a GOPTIONS statement) can be used to override the value of a style attribute.

Table 9.7 Style Elements and Attributes in the NETDRAW Style

Element/Attributes	Description	NETDRAW Option	GOPTION
GraphColors	Colors of various graph features		
gdata1	Noncritical nodes or nodes in the first zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata2	Critical nodes or nodes in the second zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata3	Nodes in the third zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata4	Nodes in the fourth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata5	Nodes in the fifth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata6	Nodes in the sixth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata7	Nodes in the seventh zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata8	Nodes in the eighth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata9	Nodes in the ninth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata10	Nodes in the tenth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata11	Nodes in the eleventh zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gdata12	Nodes in the twelfth zone	PATTERN=, ZONEPAT	CPATTERN=, COLORS=
gaxis	Borderlines		COLORS=
greferencelines	Horizontal and vertical reference lines		COLORS=
gtext	Text		CTEXT=
gtextt	Title		CTITLE=
gcdata	Arcs		COLORS=
GraphFonts	Fonts for various graph features		
GraphDataFont	Default		FTEXT=
GraphLabelFont	Annotation text		FTEXT=
GraphTitleFont	Title text		FTITLE=
GraphAxisLines	Attributes related to graph axes		
Color	GraphColors('gaxis')	CAXIS=	COLORS=
GraphConnectLine	Attributes related to arcs		
Color	GraphColors('gcdata')	CARCS=, CCRITARCS=	COLORS=
GraphReference	Attributes related to horizontal and vertical reference lines		
Color	GraphColors('greferencelines')	CREF=, CREFBRK=	COLORS=
GraphDataText	Attributes related to general text		
Color	GraphColors('gtext')	CTEXT=	CTEXT=
Font	GraphFonts('GraphDataFont')	FONT=	FTEXT=

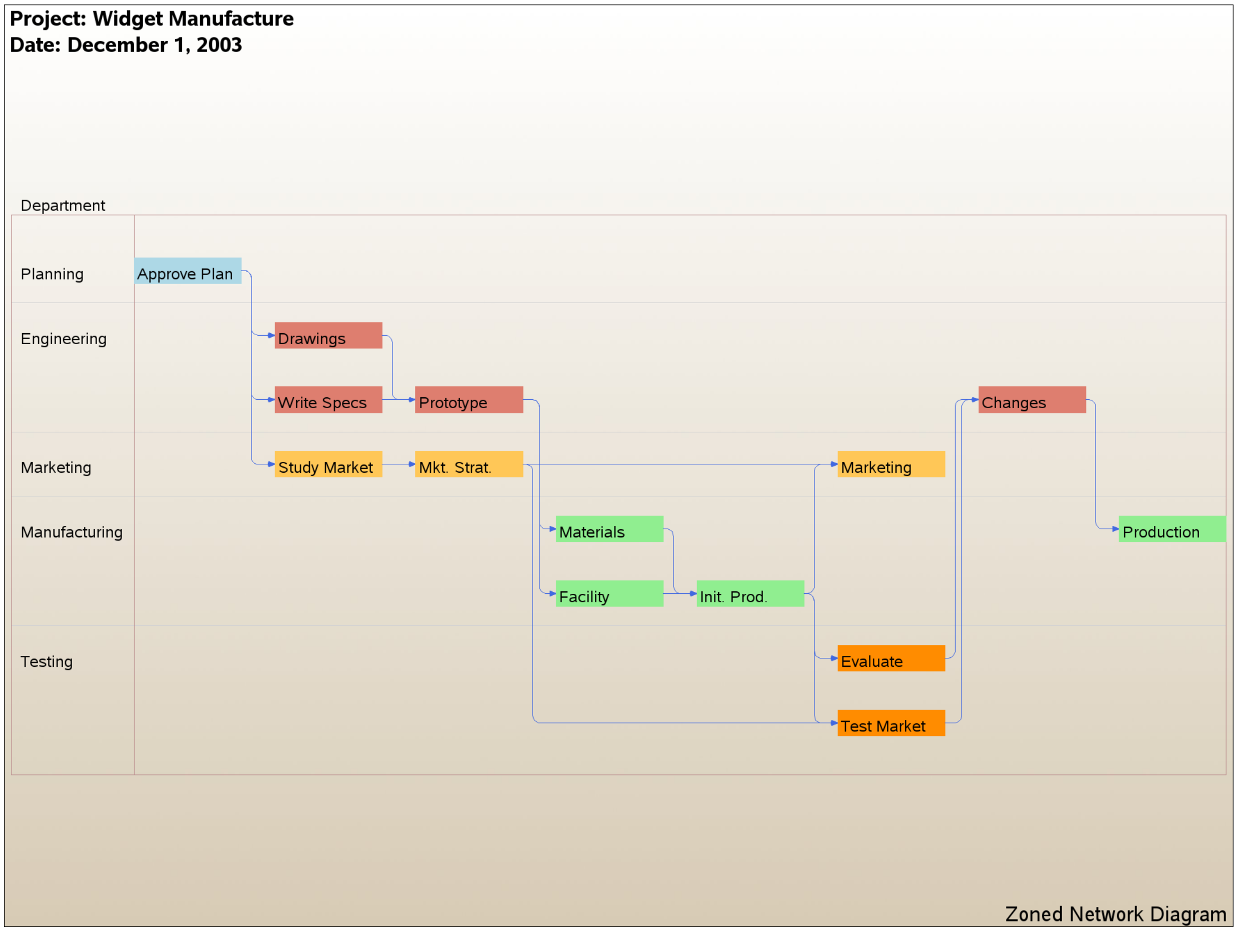
Table 9.7 (continued)

Element/Attributes	Description	NETDRAW Option	GOPTION
GraphTitleText	Attributes related to title text		
Color	GraphColors('gtextt')		CTITLE=
Font	GraphFonts('GraphTitleFont')		FTITLE=
GraphLabelText	Attributes related to annotation text		
Color	GraphColors('glabel')		CTEXT=
Font	GraphFonts('GraphLabelFont')		FTEXT=
GraphDataDefault	Default values for the attributes specified in Table 9.8		
Color	GraphColors('gdata')		COLORS=
GraphBackground	Attributes related to graph background		
Image	Background image		CBACK=

Attributes that you do not override retain the values specified in the style template.

Figure 9.9 demonstrates features of the NETDRAW graphical style. The diagram in the figure is the first output from Example 9.11.

Figure 9.9 NETDRAW Style Template: Example



Default Values

If the SAS system option GSTYLE is in effect (this is the default), then the default values of certain PROC NETDRAW options can depend on the current ODS style template. Table 9.8 lists these PROC NETDRAW options and lists the order in which PROC NETDRAW searches for each option's default value. The order assumes that the GSTYLE system option is in effect; if that is not the case, then the steps that refer to ODS style templates are ignored. Names with arguments indicate style elements and attributes of the current ODS style template. For example, "GraphAxisLines('Color')" refers to the Color attribute of the GraphAxisLines element.

Table 9.8 PROC NETDRAW Options: Search Orders for Default Values

Option	Search Order for Default Value
CARCS=	1. GraphConnectLine(Color) 2. The fourth color in the COLORS= list in the GOPTIONS statement
CAXIS=	1. GraphAxisLines(Color) 2. GraphDataDefault(Color) 3. The first color in the COLORS= list in the GOPTIONS statement
CREF=	1. GraphReference(Color) 2. GraphDataDefault(Color) 3. The first color in the COLORS= list in the GOPTIONS statement
CREFBRK=	1. GraphReference(Color) 2. GraphDataDefault(Color) 3. The first color in the COLORS= list in the GOPTIONS statement
CTEXT=	1. The value specified for the CTEXT= option in the GOPTIONS statement 2. GraphDataText(Color) 3. GraphDataDefault(Color) 4. The first color in the COLORS= list in the GOPTIONS statement
FONT=	1. The value specified for the FTEXT= option in the GOPTIONS statement 2. GraphDataText(Font) 3. The default hardware font for the graphics output device

Examples: NETDRAW Procedure

This section contains 18 examples that illustrate several features of the NETDRAW procedure. Most of the examples use the data from the Widget Manufacturing Project described in Chapter 4, "The CPM Procedure." Two tables, Table 9.9 and Table 9.10, at the end of this section list all the examples in this chapter and the options and statements in the NETDRAW procedure that are illustrated by each example.

Example 9.1: Line-Printer Network Diagram

This example uses the data set WIDGET that was used in Example 4.2 in Chapter 4, "The CPM Procedure," to illustrate the Activity-on-Node representation of the project. The following program invokes PROC NETDRAW twice. First, the activity data set WIDGET is used as input to the procedure. The activity and successor information is identified using the ACTIVITY= and SUCCESSOR= options in the ACTNET statement. The LINEPRINTER option is specified, producing the line-printer network diagram shown in Output 9.1.1.

```
data widget;
   format task $12. succ1-succ3 $12.;
   input task & days succ1 & succ2 & succ3 & ;
   datalines;
Approve Plan   5  Drawings      Study Market  Write Specs
Drawings      10  Prototype     .             .
Study Market   5  Mkt. Strat.   .             .
Write Specs    5  Prototype     .             .
Prototype     15  Materials     Facility      .
Mkt. Strat.   10  Test Market   Marketing     .
Materials     10  Init. Prod.   .             .
Facility      10  Init. Prod.   .             .
Init. Prod.   10  Test Market   Marketing     Evaluate
Evaluate      10  Changes       .             .
Test Market   15  Changes       .             .
Changes        5  Production    .             .
Production     0  .             .             .
Marketing      0  .             .             .
;

title 'Widget Manufacture';
options ps=32 ls=78;
proc netdraw data=widget lineprinter;
   actnet / activity=task successor=(succ1 succ2 succ3);
   run;
```

Output 9.1.1 Line-Printer Network Diagram

Widget Manufacture

```
                   --------------                        --------------
                -->|Drawings    |---                  -->|Materials   |---
                |  --------------  |                  |  --------------  |
                |                  |                  |                  |
                |                  |                  |                  |
                |                  |                  |------------------+--
                |                  |                  |                  |
                |                  |                  |                  |
--------------  |  --------------  |  --------------  |  --------------  |
|Approve Plan|--+->|Write Specs |---->|Prototype   |--+->|Facility    |---->
--------------  |  --------------     --------------  |  --------------
                |                                     |
                |                                     |
                |                                     |
                |                                     |
                |                                     |
                |  --------------     --------------  |
                -->|Study Market|---->|Mkt. Strat. |------------------------
                   --------------     --------------
```

Output 9.1.1 *continued*

Widget Manufacture

```
                   --------------
                -->|Test Market |---
                |  --------------  |
                |                  |
                |                  |
----------------|                  |
                |                  |
                |                  |
--------------  |  --------------  |  --------------     --------------
|Init. Prod. |--+->|Evaluate    |---->|Changes     |---->|Production  |
--------------  |  --------------     --------------     --------------
                |
                |
                |
                |
                |
                |  --------------
----------------->|Marketing   |
                   --------------
```

Next, PROC CPM is invoked to schedule the project, and the resulting Schedule data set is used as input to the NETDRAW procedure. In addition to the ACTIVITY= and SUCCESSOR= options, the DURATION= option is used in the ACTNET statement. The DURATION= option adds the values of the DURATION variable within each node of the network. The procedure also displays the values of the E_START, E_FINISH, L_START, L_FINISH, T_FLOAT, and F_FLOAT variables within each node. The network is displayed in Output 9.1.2.

```
proc cpm data=widget out=sched
         date='1dec03'd;
   activity task;
   successor succ1 succ2 succ3;
   duration days;
   run;

options ps=45 ls=90;
title2 'Schedule Information';
proc netdraw data=sched lineprinter;
   actnet / activity=task
            successor=(succ1 succ2 succ3)
            duration = days;
   run;
```

Output 9.1.2 Project Schedule

Widget Manufacture
Schedule Information

```
                              -------------------------
                              |Drawings         Dur:10|
                           -->|ES:06DEC03   EF:15DEC03|---                             -->
                           |  |LS:06DEC03   LF:15DEC03|  |                             |
                           |  |TF: 0        FF: 0     |  |                             |
                           |  -------------------------  |                             |
                           |                             |                             |
                           |                             |                             |
                           |                             |                             |--
                           |                             |                             |
                           |                             |                             |
-------------------------  |  -------------------------  |  -------------------------  |
|Approve Plan     Dur: 5|  |  |Write Specs      Dur: 5|  |  |Prototype        Dur:15|  |
|ES:01DEC03   EF:05DEC03|--+->|ES:06DEC03   EF:10DEC03|---->|ES:16DEC03   EF:30DEC03|--+->
|LS:01DEC03   LF:05DEC03|  |  |LS:11DEC03   LF:15DEC03|     |LS:16DEC03   LF:30DEC03|  |
|TF: 0        FF: 0     |  |  |TF: 5        FF: 5     |     |TF: 0        FF: 0     |  |
-------------------------  |  -------------------------     -------------------------  |
                           |                                                           |
                           |                                                           |
                           |                                                           |
                           |                                                           |
                           |                                                           |
                           |  -------------------------     -------------------------  |
                           |  |Study Market     Dur: 5|     |Mkt. Strat.      Dur:10|  |
                           -->|ES:06DEC03   EF:10DEC03|---->|ES:11DEC03   EF:20DEC03|-----
                              |LS:05JAN04   LF:09JAN04|     |LS:10JAN04   LF:19JAN04|
                              |TF:30        FF: 0     |     |TF:30        FF:30     |
                              -------------------------     -------------------------
```

Output 9.1.2 *continued*

Widget Manufacture
Schedule Information

```
-------------------------                                   -------------------------
|Materials        Dur:10|                                   |Test Market      Dur:15|
|ES:31DEC03   EF:09JAN04|---                             -->|ES:20JAN04   EF:03FEB04|---
|LS:31DEC03   LF:09JAN04|  |                             |  |LS:20JAN04   LF:03FEB04|  |
|TF: 0        FF: 0     |  |                             |  |TF: 0        FF: 0     |  |
-------------------------  |                             |  -------------------------  |
                           |                             |                             |
                           |                             |                             |
---------------------------+-----------------------------|                             |
                           |                             |                             |
                           |                             |                             |
-------------------------  |  -------------------------  |  -------------------------  |
|Facility         Dur:10|  |  |Init. Prod.      Dur:10|  |  |Evaluate         Dur:10|  |
|ES:31DEC03   EF:09JAN04|---->|ES:10JAN04   EF:19JAN04|--+->|ES:20JAN04   EF:29JAN04|---->
|LS:31DEC03   LF:09JAN04|     |LS:10JAN04   LF:19JAN04|  |  |LS:25JAN04   LF:03FEB04|
|TF: 0        FF: 0     |     |TF: 0        FF: 0     |  |  |TF: 5        FF: 5     |
-------------------------     -------------------------  |  -------------------------
                                                         |
                                                         |
                                                         |
                                                         |
                                                         |
                                                         |  -------------------------
                                                         |  |Marketing        Dur: 0|
----------------------------------------------------------->|ES:20JAN04   EF:20JAN04|
                                                            |LS:09FEB04   LF:09FEB04|
                                                            |TF:20        FF:20     |
                                                            -------------------------
```

Output 9.1.2 *continued*

Widget Manufacture
Schedule Information

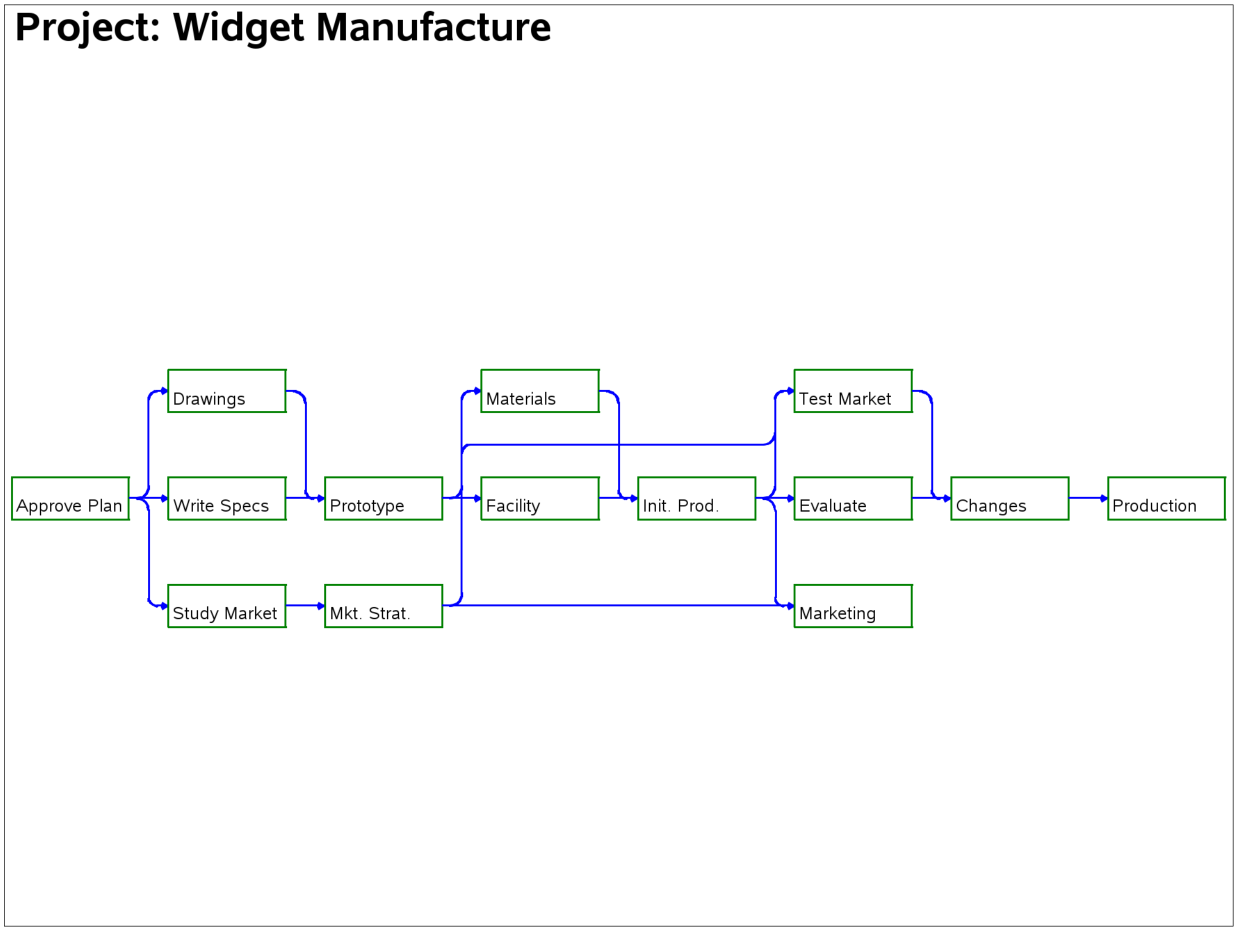
```
-------------------------     ------------------------
|Changes          Dur: 5|     |Production       Dur: 0|
|ES:04FEB04   EF:08FEB04|---->|ES:09FEB04   EF:09FEB04|
|LS:04FEB04   LF:08FEB04|     |LS:09FEB04   LF:09FEB04|
|TF: 0        FF: 0     |     |TF: 0        FF: 0     |
-------------------------     ------------------------
```

Example 9.2: Graphics Version of PROC NETDRAW

The same network used in Example 9.1 is drawn here with the GRAPHICS option (which is also the default mode of output for the NETDRAW procedure). In this example, the network is drawn before scheduling the project with PROC CPM. The global options HPOS= and VPOS= are set to 100 and 70, respectively. These options control the number of character cell positions on the screen. The network is displayed in Output 9.2.1. Note that all the nodes are the same color as specified by the PATTERN statement, the color of the arcs is blue as specified by the CARCS= option, the width of all lines is 3 as specified by the LWIDTH= option.

```
goptions hpos=100 vpos=70 border;
pattern1 c=green v=e;
title h=3  j=l ' Project: Widget Manufacture';
proc netdraw data=widget graphics;
   actnet / act=task succ=(succ1 succ2 succ3)
            height=1.5 compress
            carcs=blue lwidth=3;
   run;
```

Output 9.2.1 Project Network



Example 9.3: Spanning Multiple Pages

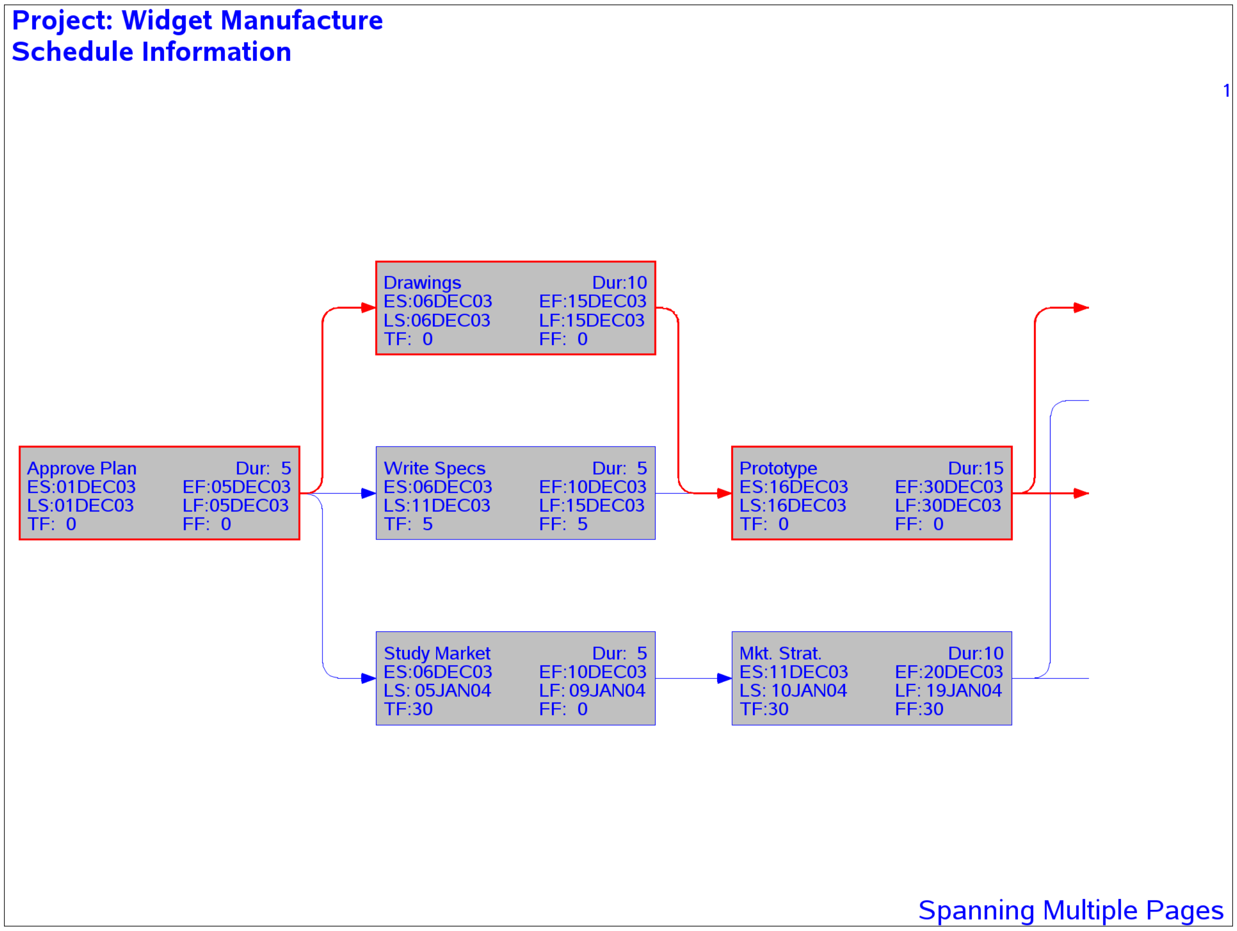
In this example, the Schedule data set produced by PROC CPM is displayed in a graphics network. As in the second part of Example 9.1, the procedure displays the duration as well as the early and late start and finish times and the total float and free float of each activity in the node corresponding to that activity. The network cannot fit on one page and is drawn across three pages, as shown in Output 9.3.1.

This example also illustrates several options for controlling the appearance of the network diagram. The pattern statements set a background fill color for all the nodes of the network (PATTERN1 and PATTERN2). The COUTLINE= and CCRITOUT= options set the outline colors for noncritical and critical activities, respectively. Recall that the procedure uses the values of the E_FINISH and L_FINISH variables to determine if an activity is critical. The CARCS= and CCRITARCS= options color the regular arcs blue and the critical arcs (arcs connecting critical activities) red, respectively and the CTEXT= options sets the color of the text to blue. Finally, the LWIDTH= option sets the default width for all the lines in the network, and the LWCRIT= option further highlights the critical arcs by drawing them with thicker lines.

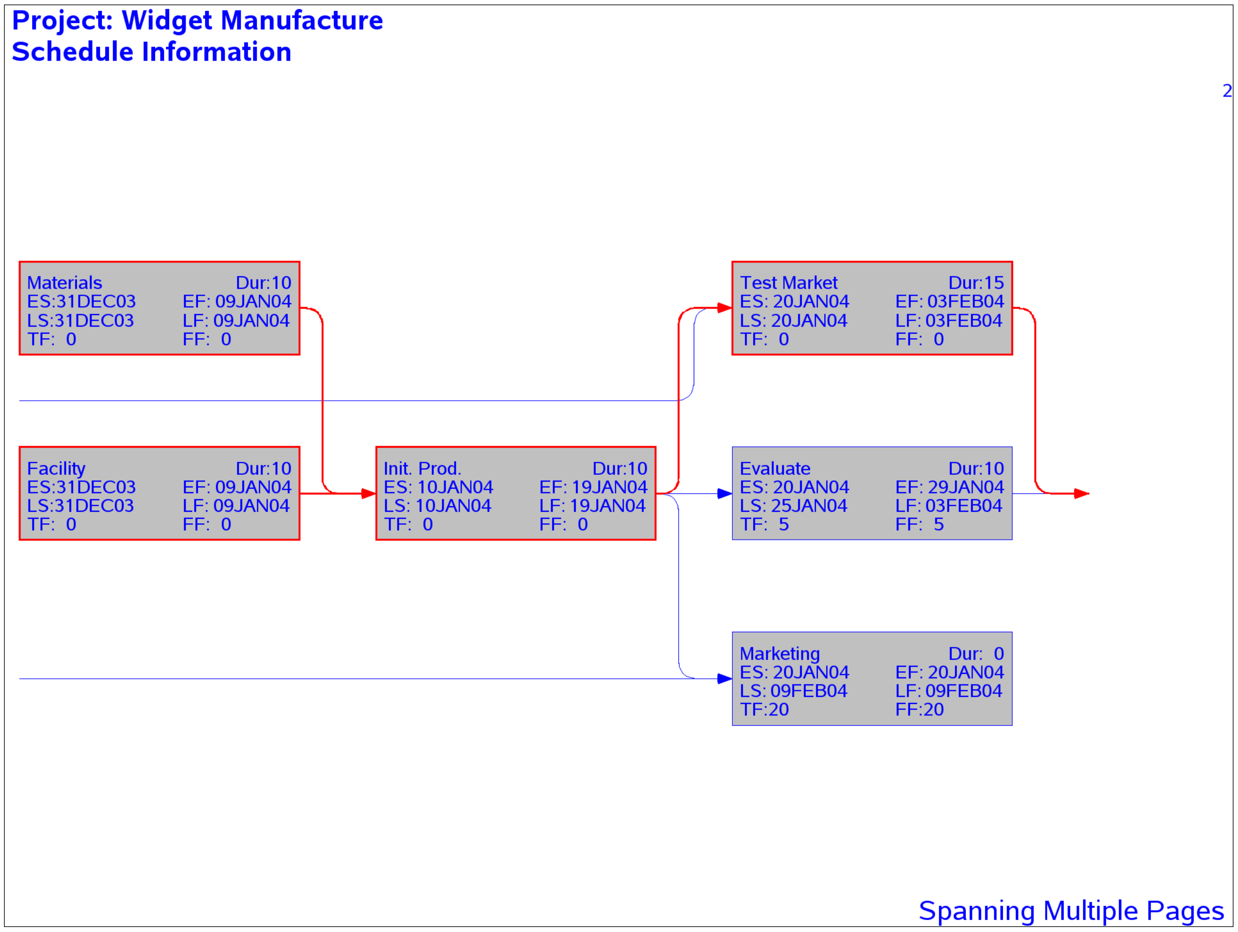
In this invocation of PROC NETDRAW, the SEPARATEARCS option is used so that the two parallel arcs leading into the activity 'Test Market' (one from 'Mkt.Strat.' and the other from 'Init. Prod.') are drawn along separate tracks instead of along a single track as in Example 9.2.

```
 /* Activity-on-Node representation of the project */
data widget;
   format task $12. succ1-succ3 $12.;
   input task & days succ1 & succ2 & succ3 & ;
   datalines;
Approve Plan   5  Drawings      Study Market  Write Specs
Drawings      10  Prototype     .             .
Study Market   5  Mkt. Strat.   .             .
Write Specs    5  Prototype     .             .
Prototype     15  Materials     Facility      .
Mkt. Strat.   10  Test Market   Marketing     .
Materials     10  Init. Prod.   .             .
Facility      10  Init. Prod.   .             .
Init. Prod.   10  Test Market   Marketing     Evaluate
Evaluate      10  Changes       .             .
Test Market   15  Changes       .             .
Changes        5  Production    .             .
Production     0  .             .             .
Marketing      0  .             .             .
;

goptions hpos=80 vpos=50 border;
pattern1 c=ltgray v=s;
pattern2 c=ltgray v=s;

title c=blue j=l h=1.5 ' Project: Widget Manufacture';
title2 c=blue j=l h=1.5 ' Schedule Information';
footnote c=blue j=r h=1.5 'Spanning Multiple Pages ';
proc netdraw data=sched graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            dur = days
            coutline=blue
            ccritout=red
            carcs=blue
            ccritarcs=red
            ctext=blue
            lwidth=1
            lwcrit=2
            separatearcs;
   run;
```

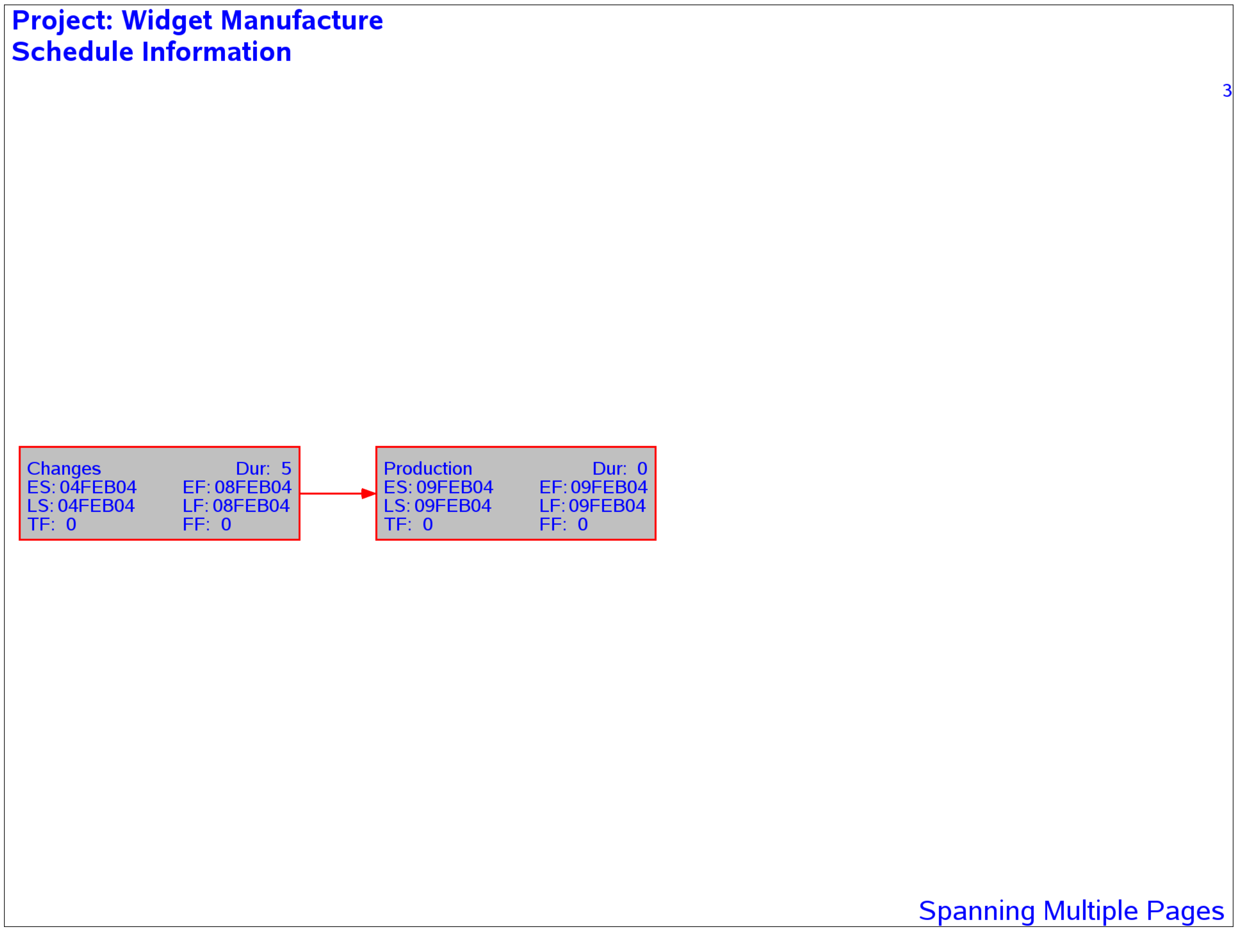
Output 9.3.1 Project Schedule


Project: Widget Manufacture
Schedule Information
1
Drawings Dur:10 ES:06DEC03 EF:15DEC03 LS:06DEC03 LF:15DEC03 TF: 0 FF: 0
Approve Plan Dur: 5 ES:01DEC03 EF:05DEC03 LS:01DEC03 LF:05DEC03 TF: 0 FF: 0
Write Specs Dur: 5 ES:06DEC03 EF:10DEC03 LS:11DEC03 LF:15DEC03 TF: 5 FF: 5
Prototype Dur:15 ES:16DEC03 EF:30DEC03 LS:16DEC03 LF:30DEC03 TF: 0 FF: 0
Study Market Dur: 5 ES:06DEC03 EF:10DEC03 LS: 05JAN04 LF: 09JAN04 TF:30 FF: 0
Mkt. Strat. Dur:10 ES:11DEC03 EF:20DEC03 LS: 10JAN04 LF: 19JAN04 TF:30 FF:30
Spanning Multiple Pages

Output 9.3.1 *continued*



Output 9.3.1 *continued*

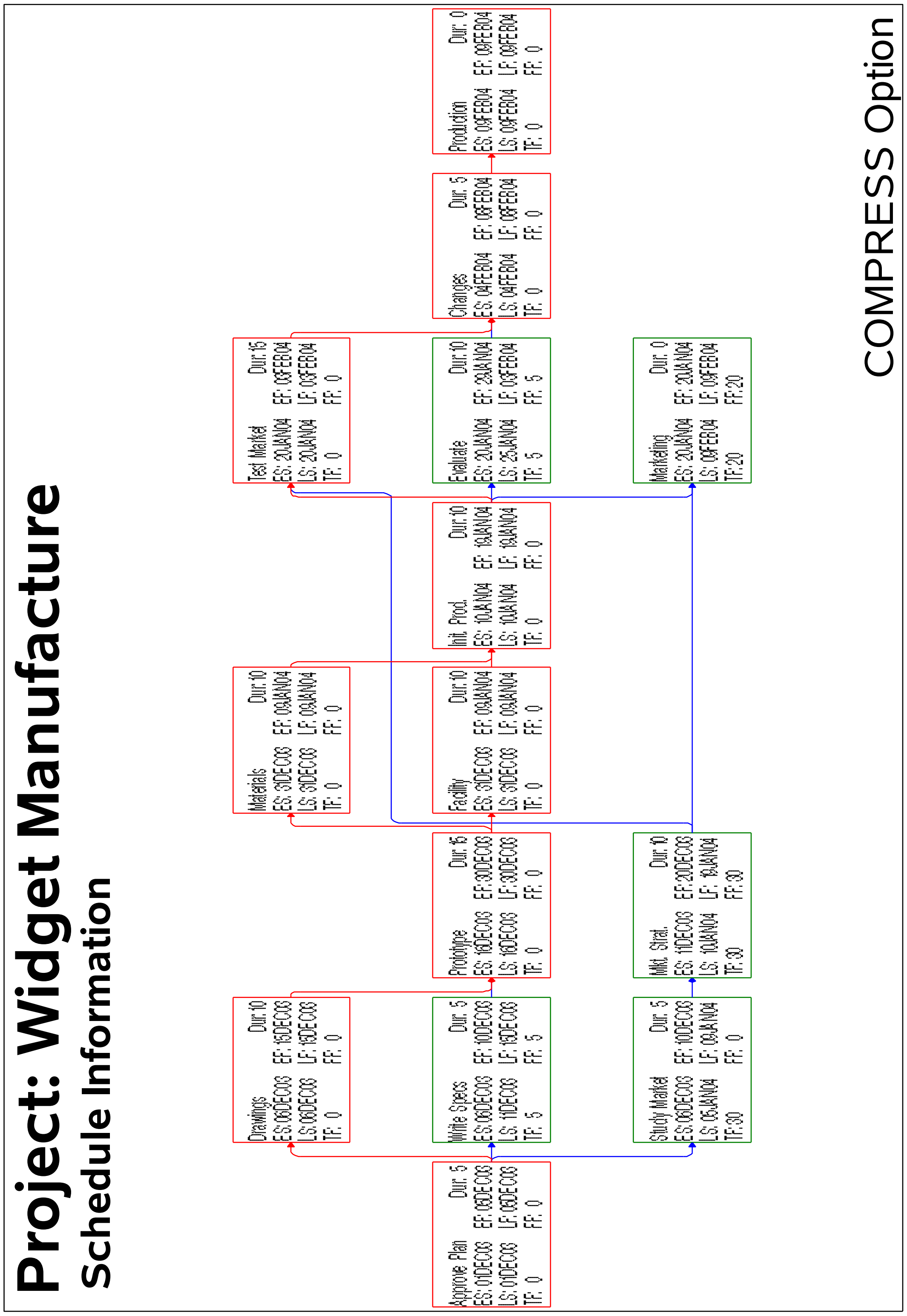


Example 9.4: The COMPRESS and PCOMPRESS Options

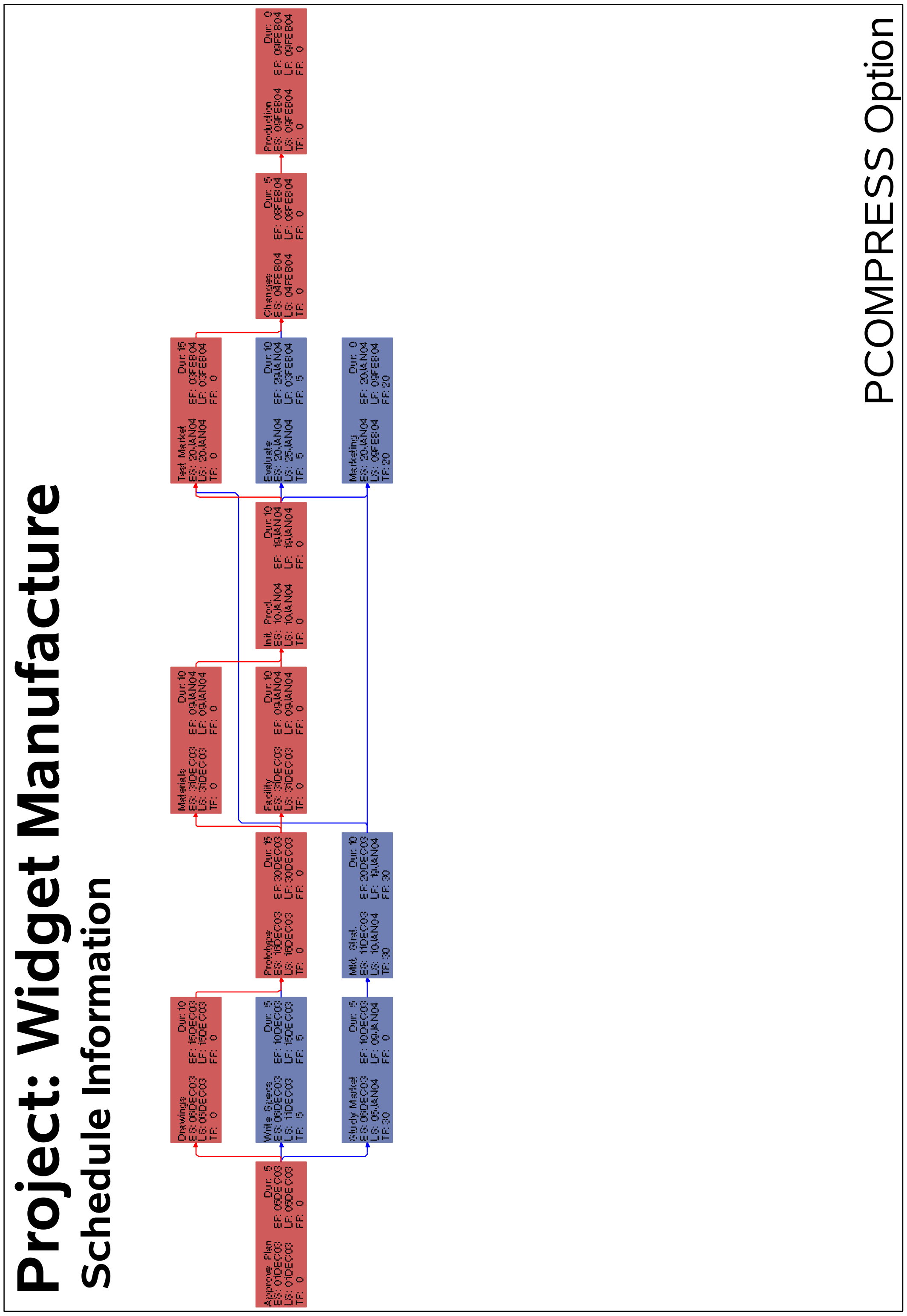
In Example 9.2, the number of character cell positions were specified so that the entire network fit on one page; in Example 9.3, the network spanned several pages. To force the network diagram to fit on one page automatically, you can use the COMPRESS or PCOMPRESS options. Both options use window transformations to fit the network on one page: the COMPRESS option treats the horizontal and vertical transformations independently of each other so that one direction may not be as compressed as the other; the PCOMPRESS option uses a proportional transformation so that each direction is compressed as much as the other. The following two invocations of PROC NETDRAW illustrate the differences in the diagrams produced.

In both network diagrams, PATTERN statements are used to color the nodes red or green, depending on whether the activities they represent are critical or not. The first PATTERN statement is for nodes corresponding to noncritical activities, and the second statement is for critical activities. The second invocation of PROC NETDRAW also uses the NOVCENTER option in the ACTNET statement to turn off centering in the vertical direction, so that the network is drawn immediately below the titles.

Output 9.4.1 Project Network: COMPRESS Option


Project: Widget Manufacture
Schedule Information
COMPRESS Option
Approve Plan Dur: 5 ES: 01DEC03 EF: 06DEC03 LS: 01DEC03 LF: 06DEC03 TF: 0 FF: 0
Drawings Dur: 10 ES: 06DEC03 EF: 15DEC03 LS: 06DEC03 LF: 15DEC03 TF: 0 FF: 0
Write Specs Dur: 5 ES: 06DEC03 EF: 10DEC03 LS: 11DEC03 LF: 15DEC03 TF: 5 FF: 5
Study Market Dur: 5 ES: 06DEC03 EF: 10DEC03 LS: 05JAN04 LF: 09JAN04 TF: 30 FF: 0
Prototype Dur: 15 ES: 16DEC03 EF: 30DEC03 LS: 16DEC03 LF: 30DEC03 TF: 0 FF: 0
Mkt. Strat. Dur: 10 ES: 11DEC03 EF: 20DEC03 LS: 10JAN04 LF: 19JAN04 TF: 30 FF: 30
Materials Dur: 10 ES: 31DEC03 EF: 09JAN04 LS: 31DEC03 LF: 09JAN04 TF: 0 FF: 0
Facility Dur: 10 ES: 31DEC03 EF: 09JAN04 LS: 31DEC03 LF: 09JAN04 TF: 0 FF: 0
Init. Prod. Dur: 10 ES: 10JAN04 EF: 19JAN04 LS: 10JAN04 LF: 19JAN04 TF: 0 FF: 0
Test Market Dur: 15 ES: 20JAN04 EF: 03FEB04 LS: 20JAN04 LF: 03FEB04 TF: 0 FF: 0
Evaluate Dur: 10 ES: 20JAN04 EF: 29JAN04 LS: 25JAN04 LF: 03FEB04 TF: 5 FF: 5
Marketing Dur: 0 ES: 20JAN04 EF: 20JAN04 LS: 09FEB04 LF: 09FEB04 TF: 20 FF: 20
Changes Dur: 5 ES: 04FEB04 EF: 08FEB04 LS: 04FEB04 LF: 08FEB04 TF: 0 FF: 0
Production Dur: 0 ES: 09FEB04 EF: 09FEB04 LS: 09FEB04 LF: 09FEB04 TF: 0 FF: 0

Output 9.4.2 Project Network: PCOMPRESS Option


Project: Widget Manufacture
Schedule Information
PCOMPRESS Option
Approve Plan ES: 01DEC03 LS: 01DEC03 TF: 0 Dur: 5 EF: 05DEC03 LF: 05DEC03 FF: 0
Drawings ES: 06DEC03 LS: 06DEC03 TF: 0 Dur: 10 EF: 15DEC03 LF: 15DEC03 FF: 0
Write Specs ES: 06DEC03 LS: 11DEC03 TF: 5 Dur: 5 EF: 10DEC03 LF: 15DEC03 FF: 5
Study Market ES: 06DEC03 LS: 05JAN04 TF: 30 Dur: 5 EF: 10DEC03 LF: 09JAN04 FF: 0
Prototype ES: 16DEC03 LS: 16DEC03 TF: 0 Dur: 15 EF: 30DEC03 LF: 30DEC03 FF: 0
Mkt. Strat. ES: 11DEC03 LS: 10JAN04 TF: 30 Dur: 10 EF: 20DEC03 LF: 19JAN04 FF: 30
Materials ES: 31DEC03 LS: 31DEC03 TF: 0 Dur: 10 EF: 09JAN04 LF: 09JAN04 FF: 0
Facility ES: 31DEC03 LS: 31DEC03 TF: 0 Dur: 10 EF: 09JAN04 LF: 09JAN04 FF: 0
Init. Prod. ES: 10JAN04 LS: 10JAN04 TF: 0 Dur: 10 EF: 19JAN04 LF: 19JAN04 FF: 0
Test Market ES: 20JAN04 LS: 20JAN04 TF: 0 Dur: 15 EF: 03FEB04 LF: 03FEB04 FF: 0
Evaluate ES: 20JAN04 LS: 25JAN04 TF: 5 Dur: 10 EF: 29JAN04 LF: 03FEB04 FF: 5
Marketing ES: 20JAN04 LS: 09FEB04 TF: 20 Dur: 0 EF: 20JAN04 LF: 09FEB04 FF: 20
Changes ES: 04FEB04 LS: 04FEB04 TF: 0 Dur: 5 EF: 08FEB04 LF: 08FEB04 FF: 0
Production ES: 09FEB04 LS: 09FEB04 TF: 0 Dur: 0 EF: 09FEB04 LF: 09FEB04 FF: 0

```
goptions border;

pattern1 v=e c=green;
pattern2 v=e c=red;

title j=l h=3 ' Project: Widget Manufacture';
title2 j=l h=2 '  Schedule Information';
footnote j=r h=2 'COMPRESS Option ';
proc netdraw data=sched graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            dur = days
            carcs=blue
            ccritarcs=red
            separatearcs
            font=swiss
            height=1.5
            compress;
   run;

footnote j=r h=2 'PCOMPRESS Option ';
proc netdraw data=sched graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            dur = days
            carcs=blue
            ccritarcs=red
            separatearcs
            font=swiss
            height=1.5
            pcompress
            novcenter;
   run;
```

Example 9.5: Controlling the Display Format

In addition to the COMPRESS and PCOMPRESS options and the HPOS= and VPOS= global options, you can control the amount of information displayed on a single page by

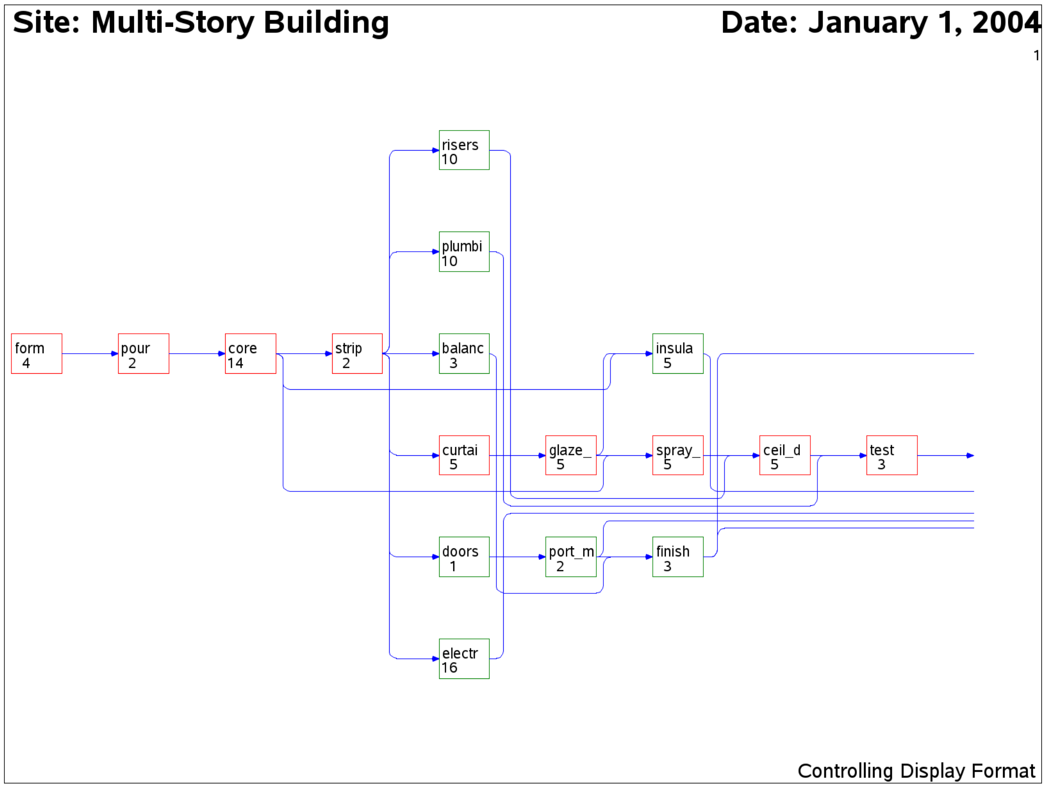
- turning off the default ID variable selection (using the NODEFID option) and using the ID= option to select only a few variables of interest in the Network data set
- setting the dimensions of each node using the BOXWIDTH= and the BOXHT= options
- specifying the horizontal and vertical distances between nodes using the XBETWEEN= and YBETWEEN= options, respectively

This example uses the data from Example 8.1 in Chapter 8, "The GANTT Procedure," and some of the options just mentioned to produce the network diagram shown in Output 9.5.1. The ID= and NODEFID options are used to display only the activity name and duration values within each node. The NOLABEL option suppresses the display of the variable names within each node. Some of the activity names are truncated by the BOXWIDTH= option. Even with the restrictions imposed, the network is too large to fit on one page and spans two pages. Note that on devices with higher resolution, you can increase the values of HPOS and VPOS (still maintaining readability) to enable more of the network to be drawn on one page.

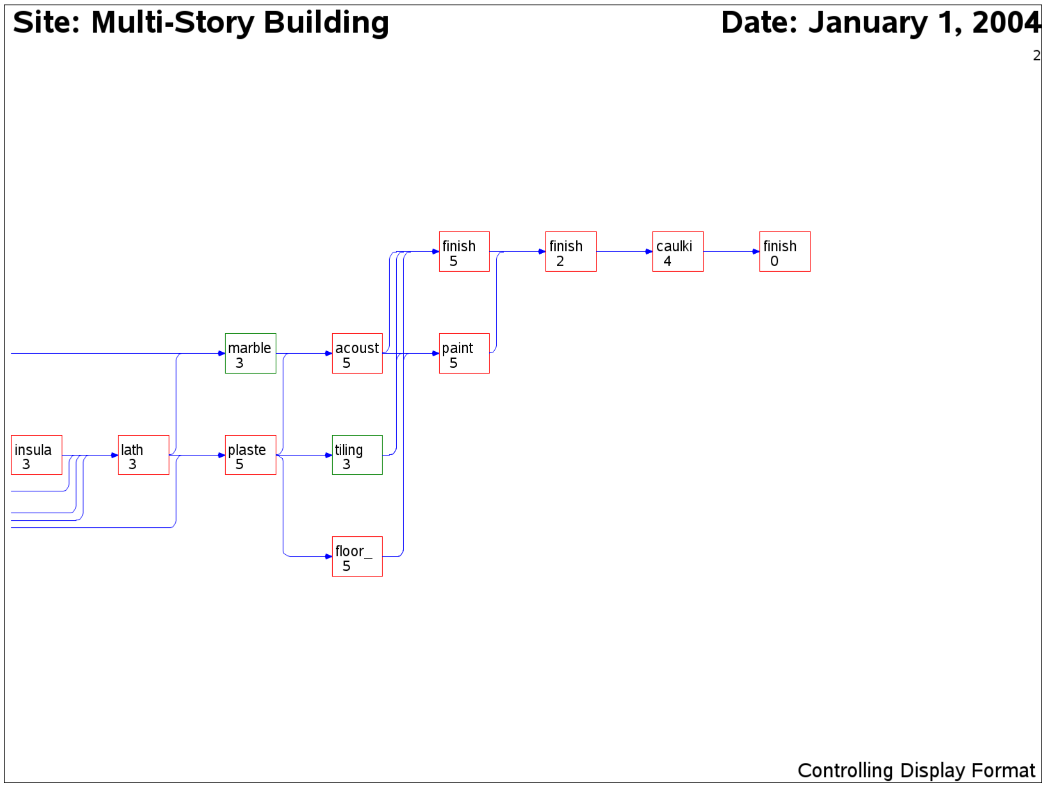
```
data ex;
   format activity $20. success1 $20. success2 $20. success3 $20.
                        success4 $20.;
   input activity dur success1-success4;
   datalines;
form                 4 pour . . .
pour                 2 core . . .
core                14 strip spray_fireproof insulate_walls .
strip                2 plumbing curtain_wall risers doors
strip                2 electrical_walls balance_elevator . .
curtain_wall         5 glaze_sash . . .
glaze_sash           5 spray_fireproof insulate_walls . .
spray_fireproof      5 ceil_ducts_fixture . . .
ceil_ducts_fixture   5 test . . .
plumbing            10 test . . .
test                 3 insulate_mechanical . . .
insulate_mechanical  3 lath . . .
insulate_walls       5 lath . . .
risers              10 ceil_ducts_fixture . . .
doors                1 port_masonry . . .
port_masonry         2 lath finish_masonry . .
electrical_walls    16 lath . . .
balance_elevator     3 finish_masonry . . .
finish_masonry       3 plaster marble_work . .
lath                 3 plaster marble_work . .
plaster              5 floor_finish tiling acoustic_tiles .
marble_work          3 acoustic_tiles . . .
acoustic_tiles       5 paint finish_mechanical . .
tiling               3 paint finish_mechanical . .
floor_finish         5 paint finish_mechanical . .
paint                5 finish_paint . . .
finish_mechanical    5 finish_paint . . .
finish_paint         2 caulking_cleanup . . .
caulking_cleanup     4 finished . . .
finished             0 . . . .
;


proc cpm finishbefore date='1jan04'd out=sched;
   activity activity;
   duration dur;
   successors success1-success4;
   run;

proc sort;
   by e_start;
   run;
```

```
pattern1 v=e c=green;
pattern2 v=e c=red;

title j=l h=3  ' Site: Multi-Story Building'
      j=r ' Date: January 1, 2004';
footnote j=r h=2 'Controlling Display Format ';

proc netdraw data=sched graphics;
   actnet / act = activity
            succ = (success1-success4)
            id = ( activity dur )
            nolabel nodefaultid
            boxwidth = 6
            ybetween = 6
            separatearcs;
run;
```

Output 9.5.1 Controlling the Display Format



Output 9.5.1 *continued*



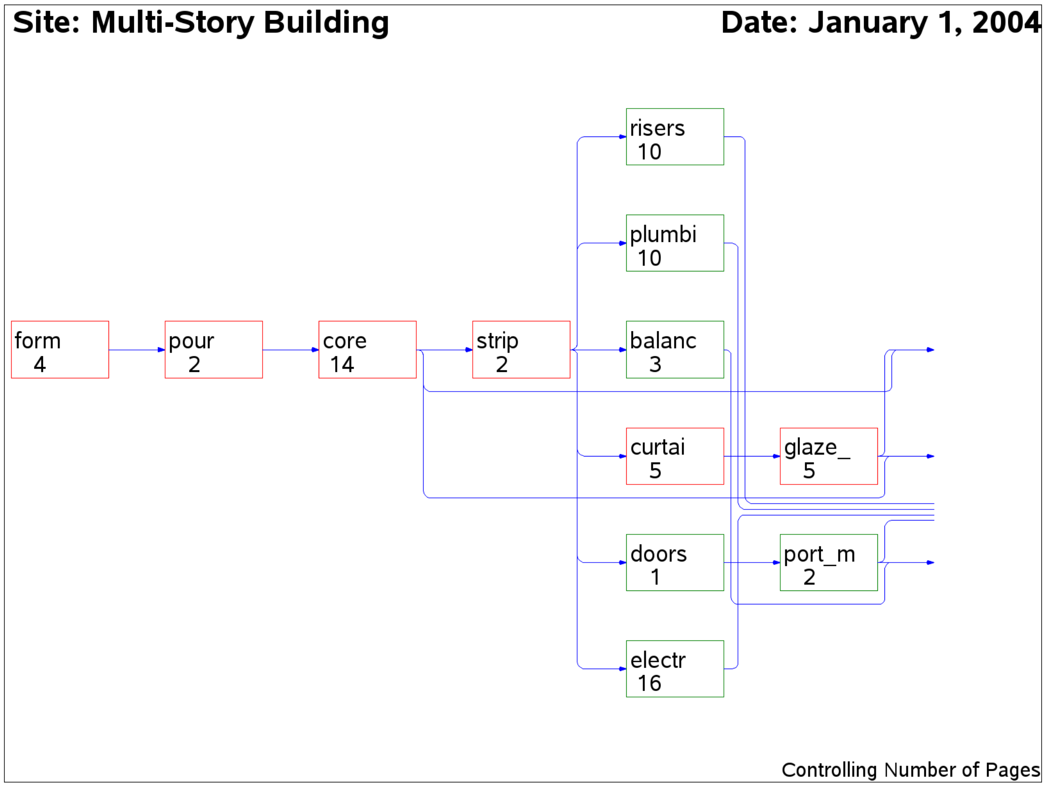
You can also control the format of the display by specifying the number of pages into which the network diagram should be split. You can do this by

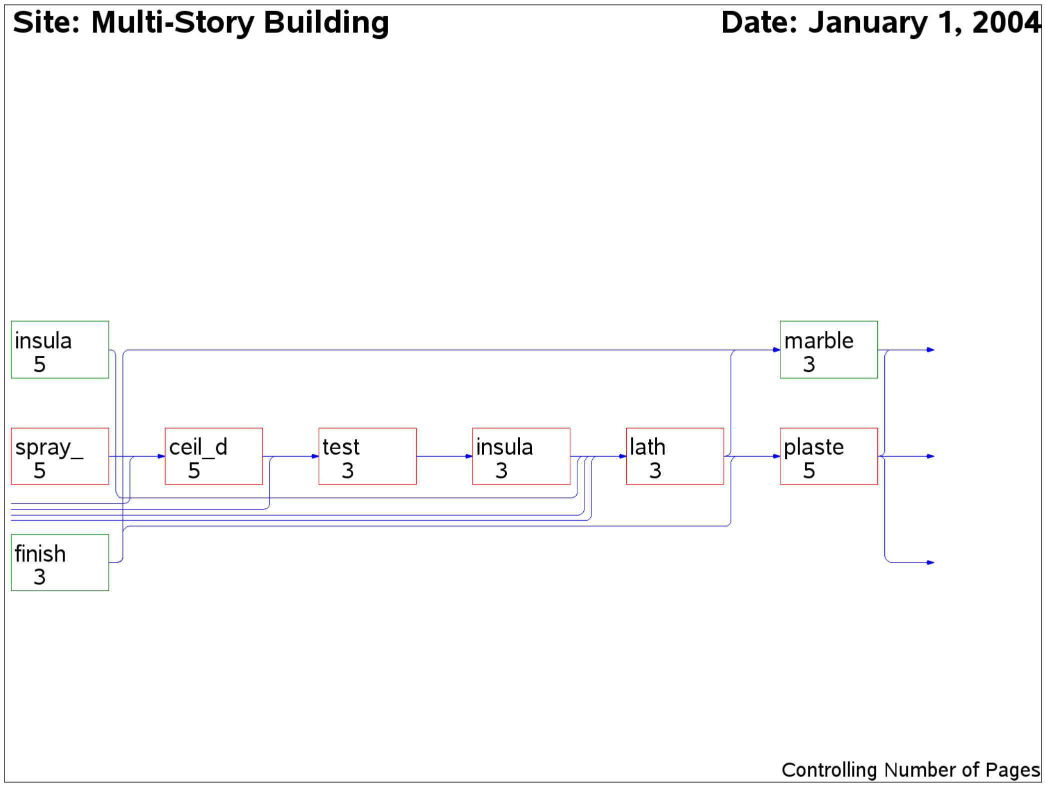
- using the HPAGES= and VPAGES= options, which specify the number of pages in the horizontal and vertical directions, respectively
- setting the number of nodes in the horizontal and vertical directions using the NXNODES= and NYNODES= options, respectively

The following statements invoke PROC NETDRAW with some additional page control options. The HTEXT= option is also used to control the height of the text used in the diagram.

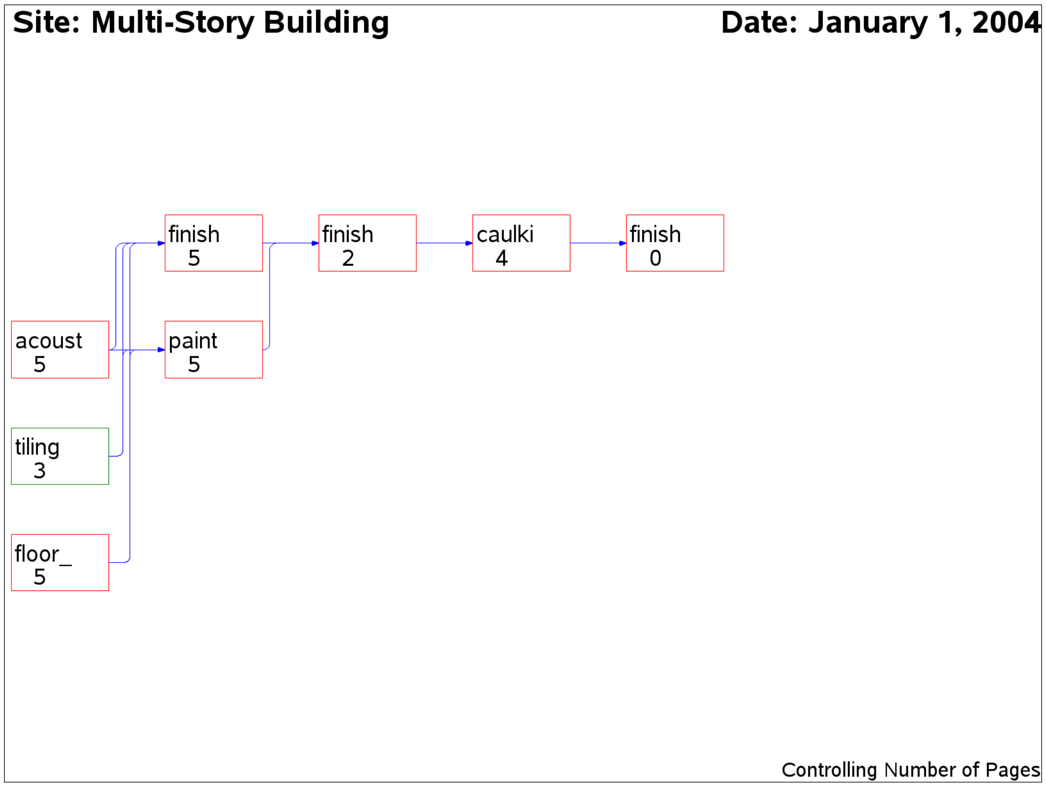
```
footnote j=r h=2 'Controlling Number of Pages';

proc netdraw data=sched graphics;
   actnet / act = activity
            succ = (success1-success4)
            id = ( activity dur )
            nolabel nodefaultid
            boxwidth = 6
            ybetween = 6
            separatearcs
            htext=2
            nopagenumber
            hpages=3 vpages=1;
run;
```

Output 9.5.2 Controlling the Number of Pages





Output 9.5.2 *continued*



Example 9.6: Nonstandard Precedence Relationships

This example illustrates the use of the LAG= option to indicate nonstandard precedence relationships between activities. Consider the widget manufacturing project described in the earlier examples. Some of the precedence constraints between the activities may be nonstandard or have a nonzero lag value. For example, the activity ‘Init. Prod.’ may not be able to start until 2 days after the completion of the activity ‘Facility.’ The Network data set is displayed in Output 9.6.1. The variable lagdur indicates the type of relationship between the activities specified in the variables task and succ.

The following statements invoke PROC NETDRAW with the LAG= option. The resulting network diagram is shown in Output 9.6.2.

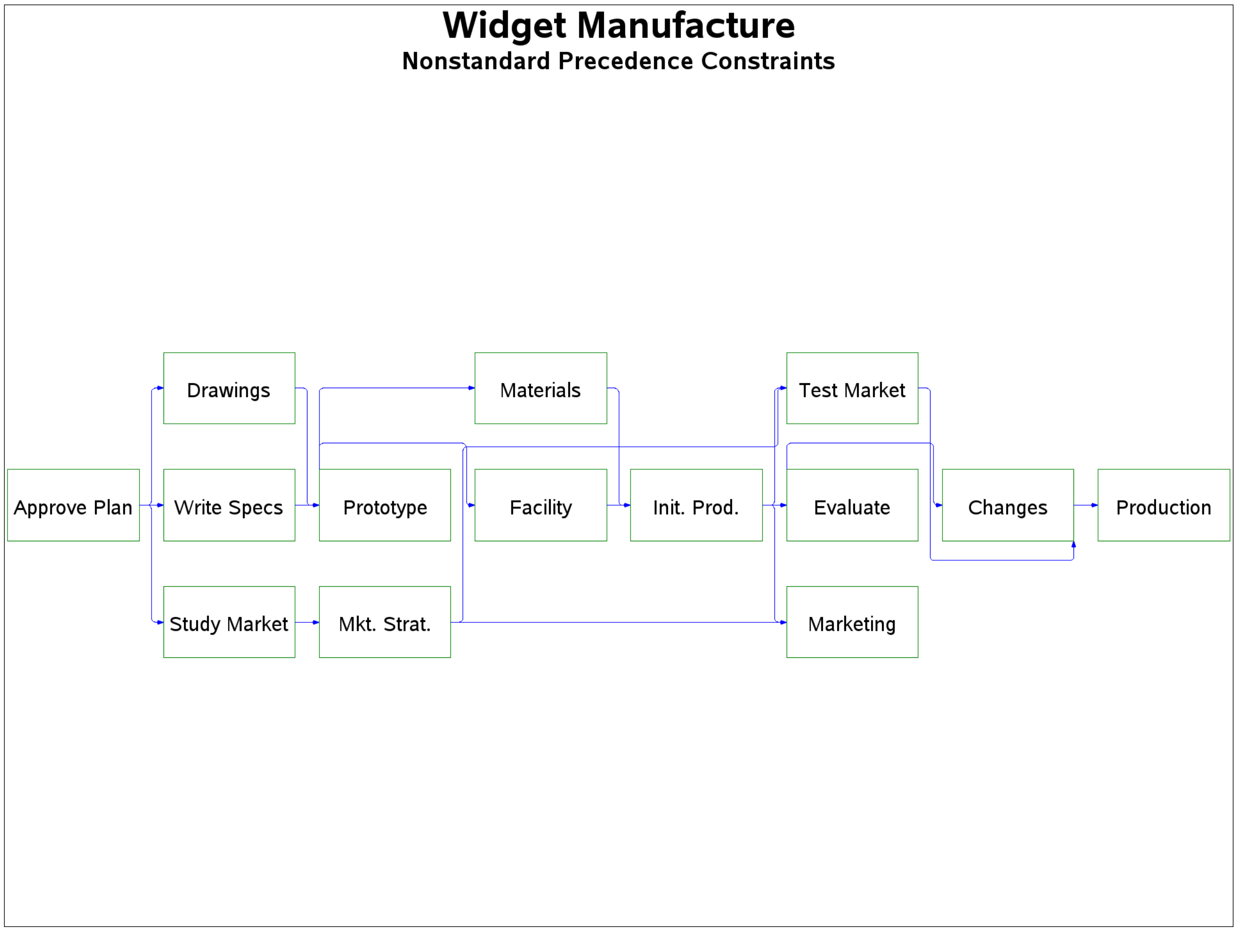
```
pattern1 v=e c=green;

title h=3 'Widget Manufacture';
title2 h=2 'Nonstandard Precedence Constraints';

proc netdraw graphics data=widglag;
   actnet / act=task
            succ=succ
            lag=lagdur
            pcompress
            htext=3 boxht=3 arrowhead=2
            xbetween=7 ybetween=9
            centerid
            separatearcs;
   run;
```

Output 9.6.1 Network with Nonstandard Precedence Constraints

Widget Manufacture
Network Data Set

Obs	task	days	succ	lagdur
1	Approve Plan	5	Drawings	
2	Approve Plan	5	Study Market	
3	Approve Plan	5	Write Specs	
4	Drawings	10	Prototype	
5	Study Market	5	Mkt. Strat.	
6	Write Specs	5	Prototype	
7	Prototype	15	Materials	ss_9
8	Prototype	15	Facility	ss_9
9	Mkt. Strat.	10	Test Market	
10	Mkt. Strat.	10	Marketing	
11	Materials	10	Init. Prod.	
12	Facility	10	Init. Prod.	fs_2
13	Init. Prod.	10	Test Market	
14	Init. Prod.	10	Marketing	
15	Init. Prod.	10	Evaluate	
16	Evaluate	10	Changes	ss_6
17	Test Market	15	Changes	ff_3
18	Changes	5	Production	
19	Production	0		
20	Marketing	0		

Output 9.6.2 Network Diagram with Nonstandard Precedence Constraints

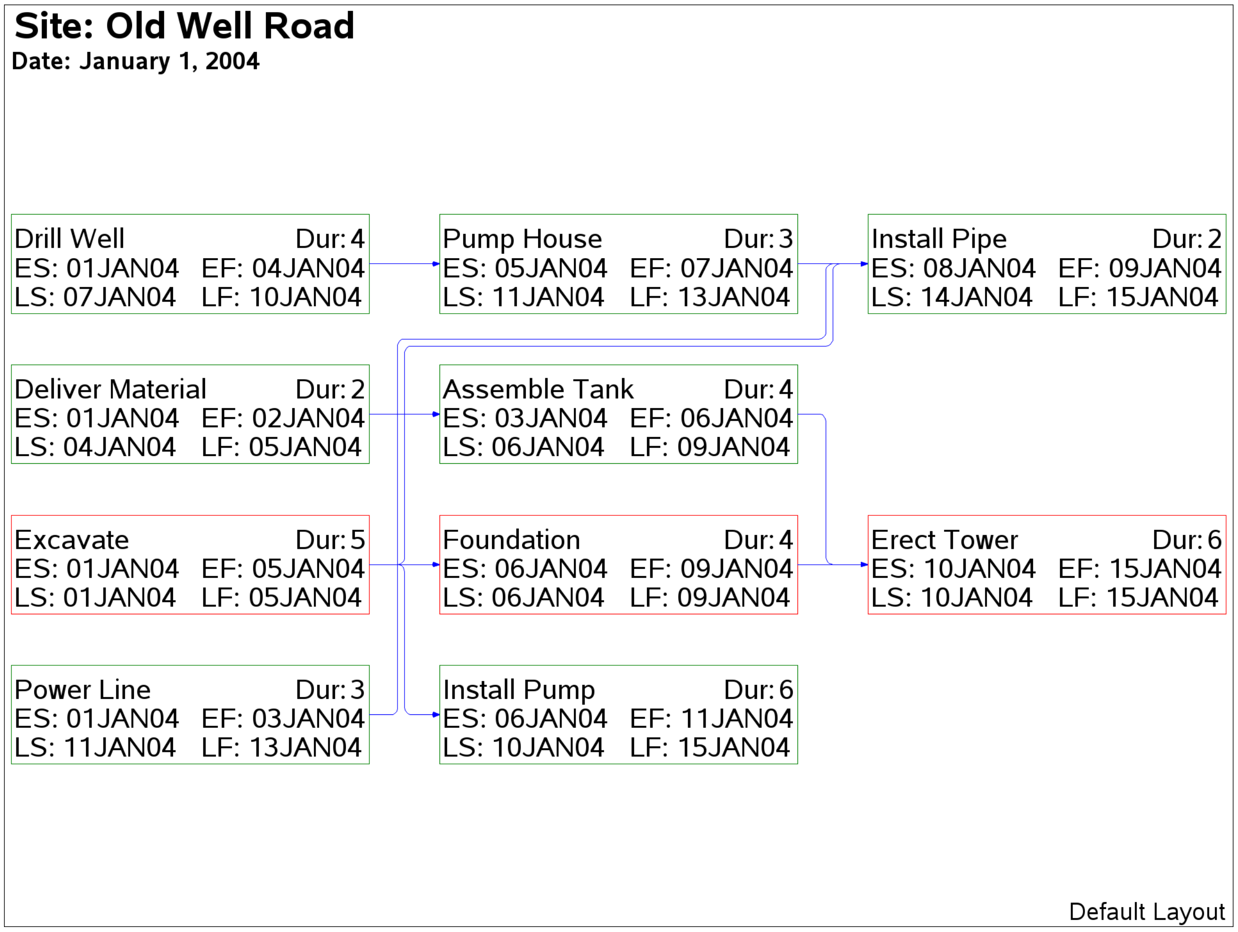


Example 9.7: Controlling the Arc-Routing Algorithm

This example illustrates the use of the DP and HTRACKS= options to control the routing of the arcs connecting the nodes. The project is a simple construction project with the following data. A Schedule data set produced by PROC CPM is input to PROC NETDRAW. The first invocation of the procedure illustrates the default layout of the network. As explained in the section "Layout of the Network" on page 702, the NETDRAW procedure uses a simple heuristic to route the arcs between the nodes. In the resulting diagram displayed in Output 9.7.1, note that the specification of BOXHT=3 limits the number of rows within each node so that the float values are not displayed.

```
data exmp1;
   input task     $ 1-16
         duration
         succesr1 $ 21-35
         succesr2 $ 36-50
         succesr3 $ 51-65;
   datalines;
Drill Well       4  Pump House
Pump House       3  Install Pipe
Power Line       3  Install Pipe
Excavate         5  Install Pipe   Install Pump   Foundation
Deliver Material 2  Assemble Tank
Assemble Tank    4  Erect Tower
Foundation       4  Erect Tower
Install Pump     6
Install Pipe     2
Erect Tower      6
;

proc cpm data=exmp1 date='1jan04'd out=sched;
   activity task;
   duration duration;
   successor succesr1 succesr2 succesr3;
   run;

pattern1 v=e c=green;
pattern2 v=e c=red;

title j=l h=3 ' Site: Old Well Road';
title2 j=l h=2 ' Date: January 1, 2004';
footnote j=r h=2 'Default Layout ';
proc netdraw data=sched graphics;
   actnet / act = task
            dur = duration
            succ = (succesr1-succesr3)
            boxht = 3 xbetween = 10
            separatearcs
            htext=2
            pcompress;
   run;
```

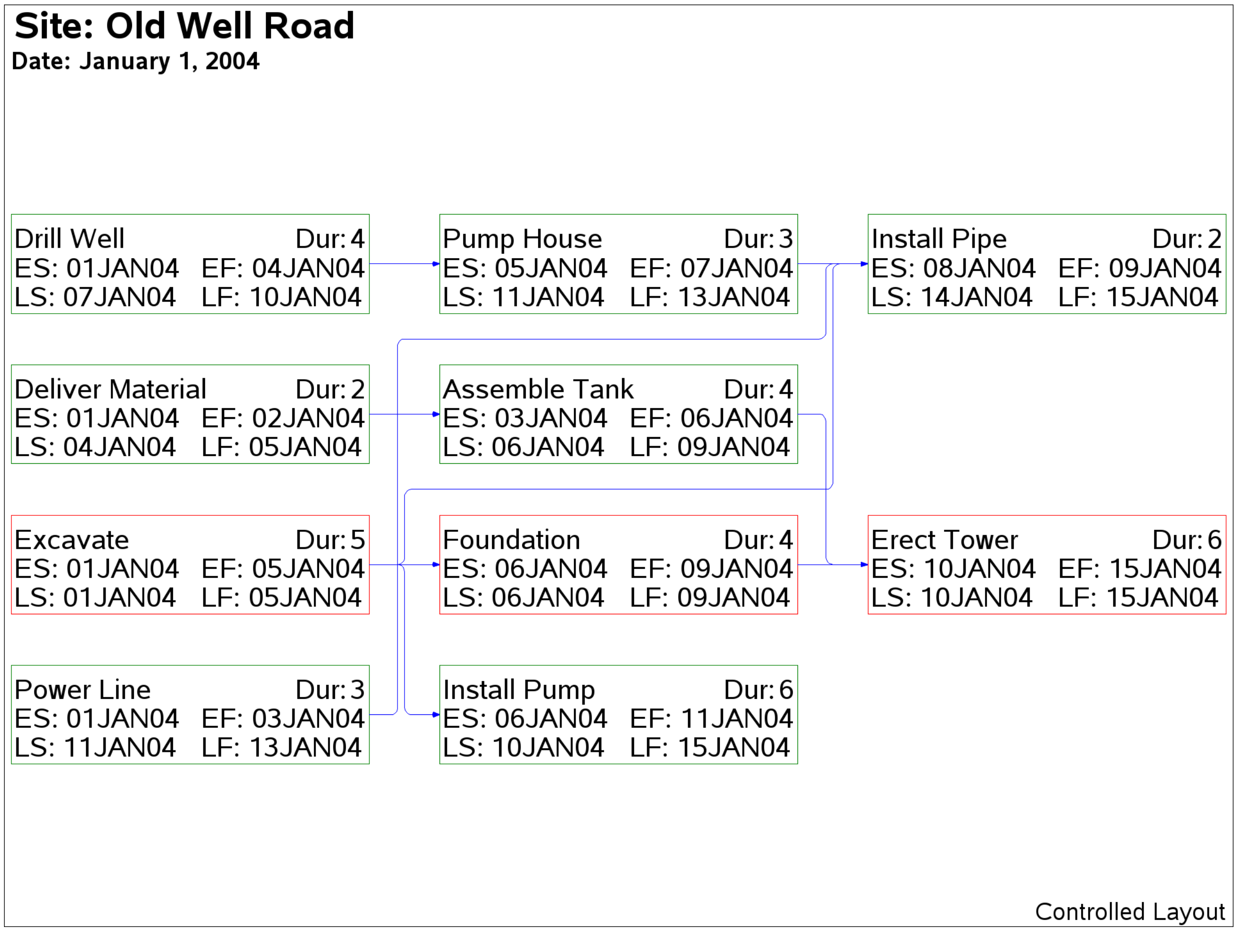
Output 9.7.1 Arc Routing: Default Layout


Site: Old Well Road
Date: January 1, 2004
Drill Well Dur: 4 ES: 01JAN04 EF: 04JAN04 LS: 07JAN04 LF: 10JAN04
Pump House Dur: 3 ES: 05JAN04 EF: 07JAN04 LS: 11JAN04 LF: 13JAN04
Install Pipe Dur: 2 ES: 08JAN04 EF: 09JAN04 LS: 14JAN04 LF: 15JAN04
Deliver Material Dur: 2 ES: 01JAN04 EF: 02JAN04 LS: 04JAN04 LF: 05JAN04
Assemble Tank Dur: 4 ES: 03JAN04 EF: 06JAN04 LS: 06JAN04 LF: 09JAN04
Excavate Dur: 5 ES: 01JAN04 EF: 05JAN04 LS: 01JAN04 LF: 05JAN04
Foundation Dur: 4 ES: 06JAN04 EF: 09JAN04 LS: 06JAN04 LF: 09JAN04
Erect Tower Dur: 6 ES: 10JAN04 EF: 15JAN04 LS: 10JAN04 LF: 15JAN04
Power Line Dur: 3 ES: 01JAN04 EF: 03JAN04 LS: 11JAN04 LF: 13JAN04
Install Pump Dur: 6 ES: 06JAN04 EF: 11JAN04 LS: 10JAN04 LF: 15JAN04
Default Layout

Next, a different routing of the arcs is obtained by specifying the DP and the HTRACKS= options. As a result of these options, the NETDRAW procedure uses a dynamic programming algorithm to route the arcs, limiting the number of horizontal tracks used to 1. The resulting network diagram is shown in Output 9.7.2. Notice that at most one arc is drawn in each horizontal track. Recall that, by default, the procedure uses a dynamic programming algorithm for arc routing if the number of tracks is restricted to be less than the maximum number of successors. Thus, for this example, the default routing option will be DP, even if it is not explicitly specified (because HTRACKS = 1 and the maximum number of successors is 3).

```
footnote j=r h=2 'Controlled Layout ';
proc netdraw data=sched graphics;
   actnet / act = task
            dur = duration
            succ = (succesr1-succesr3)
            boxht = 3 xbetween = 10
            separatearcs
            htracks=1
            htext=2
            pcompress
            dp;
   run;
```

Output 9.7.2 Arc Routing: Controlled Layout



Example 9.8: PATTERN and SHOWSTATUS Options

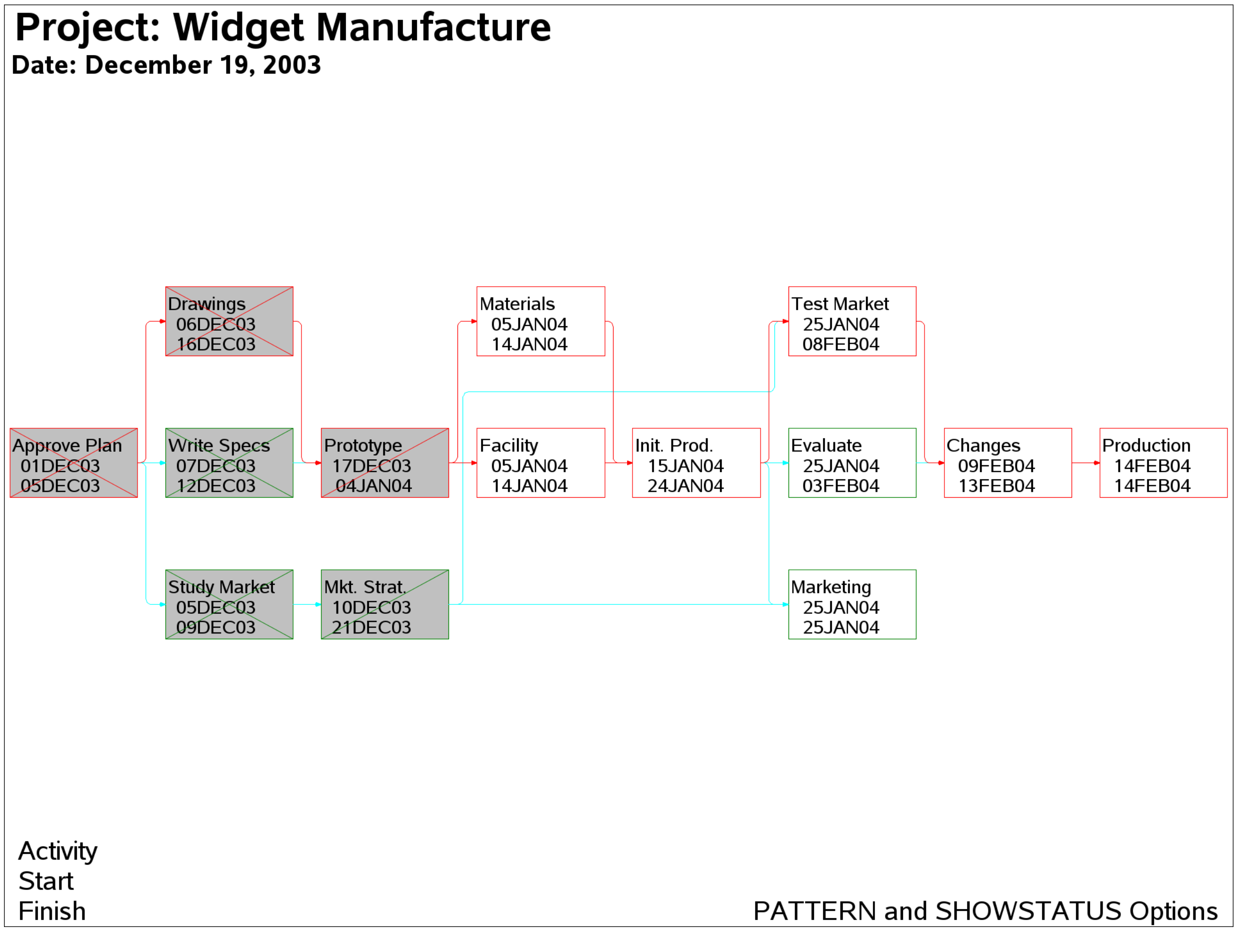
As a project progresses, in addition to the criticality of the activities, you may also want to display the status of the activity: whether it is in progress, has been completed, or is still to be scheduled. The SHOWSTATUS option in the ACTNET statement displays this additional information. In the current example, the same progress data as shown in Example 4.13 in Chapter 4, "The CPM Procedure," are used to illustrate the SHOWSTATUS option. The following program shows the necessary code. First, PROC CPM schedules the project with the SHOWFLOAT option; this enables activities that are already in progress or completed also to show nonzero float. Following this, a DATA step sets the variable style to '3' for activities that are completed or in progress; the remaining activities have missing values for this variable.

PROC NETDRAW is then invoked with the SHOWSTATUS option, which draws two diagonal lines across nodes referring to completed activities and one diagonal line for in-progress activities. The PATTERN= option in the ACTNET statement identifies the variable style containing the pattern information. Thus, the third pattern statement is used for in-progress or completed activities; the other activities (which have missing values for the variable style) use the second or the first pattern statement according to whether or not they are critical. However, since the first two PATTERN statements have EMPTY fill patterns specified, the nodes representing activities that have not yet started are in fact colored on the basis of the COUTLINE= and CCRITOUT= options. The resulting network diagram is shown in Output 9.8.1.

```
data holidays;
   format holiday holifin date7.;
   input holiday & date7. holifin & date7. holidur;
   datalines;
24dec03  26dec03  4
01jan04  .        .
;

* actual schedule at timenow = 19dec03;
data actual;
   format task $12. sdate fdate date7.;
   input task & sdate & date7. fdate & date7. pctc rdur;
   datalines;
Approve Plan  01dec03  05dec03  .    .
Drawings      06dec03  16dec03  .    .
Study Market  05dec03  .        100  .
Write Specs   07dec03  12dec03  .    .
Prototype     .        .        .    .
Mkt. Strat.   10dec03  .        .    3
Materials     .        .        .    .
Facility      .        .        .    .
Init. Prod.   .        .        .    .
Evaluate      .        .        .    .
Test Market   .        .        .    .
Changes       .        .        .    .
Production    .        .        .    .
Marketing     .        .        .    .
;

* merge the predicted information with network data;
data widgact;
   merge  actual widget;
   run;

* estimate schedule based on actual data;
proc cpm data=widgact holidata=holidays
         out=widgupd date='1dec03'd;
   activity task;
   succ     succ1 succ2 succ3;
   duration days;
   holiday  holiday / holifin=(holifin);
   actual / as=sdate af=fdate timenow='19dec03'd
            remdur=rdur pctcomp=pctc showfloat;
   run;

/* Set patterns for activities that have started */
data netin;
   set widgupd;
   if a_start ^= . then style = 3;
   run;
```

```
goptions hpos=120 vpos=70 border;
pattern1 c=green  v=e;
pattern2 c=red    v=e;
pattern3 c=ltgray v=s;

title  j=l h=3 ' Project: Widget Manufacture';
title2 j=l h=2 ' Date: December 19, 2003';
footnote1 j=l h=2 '  Activity';
footnote2 j=l h=2 '  Start';
footnote3 j=l h=2 '  Finish'
          j=r h=2 'PATTERN and SHOWSTATUS Options  ';
proc netdraw data=netin graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            ybetween = 10
            separatearcs
            pcompress
            id=(task e_start e_finish)
            nodefid nolabel
            carcs=cyan
            ccritarcs=red
            coutline = green
            ccritout = red
            showstatus
            pattern = style
            htext=2;
   run;
```

Output 9.8.1 PATTERN and SHOWSTATUS Options


Project: Widget Manufacture
Date: December 19, 2003
Approve Plan 01DEC03 05DEC03
Drawings 06DEC03 16DEC03
Write Specs 07DEC03 12DEC03
Study Market 05DEC03 09DEC03
Prototype 17DEC03 04JAN04
Mkt. Strat. 10DEC03 21DEC03
Materials 05JAN04 14JAN04
Facility 05JAN04 14JAN04
Init. Prod. 15JAN04 24JAN04
Test Market 25JAN04 08FEB04
Evaluate 25JAN04 03FEB04
Marketing 25JAN04 25JAN04
Changes 09FEB04 13FEB04
Production 14FEB04 14FEB04
Activity
Start
Finish
PATTERN and SHOWSTATUS Options

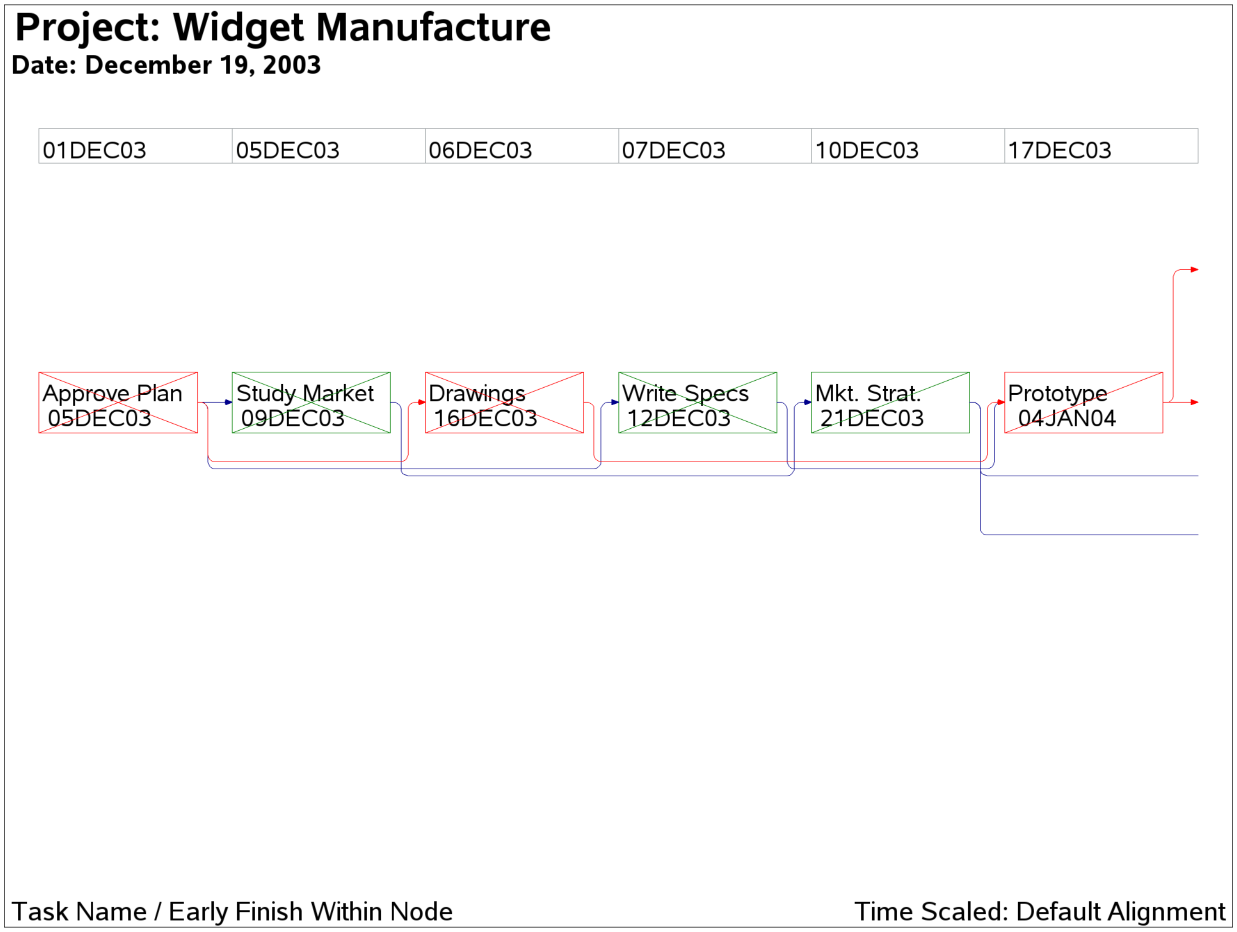
Example 9.9: Time-Scaled Network Diagram

This example illustrates the use of the TIMESCALE and ALIGN= options to draw time-scaled network diagrams. The Schedule data set WIDGUPD, produced by PROC CPM in the previous example, is used. First, PROC NETDRAW is invoked with the TIMESCALE option without any ALIGN= specification. By default, the procedure aligns the nodes to coincide with the early start times of the activities. The network spans two pages (HPAGES=2 VPAGES=1), as shown in Output 9.9.1. The HMARGIN= and VMARGIN= options add extra space around the margins.

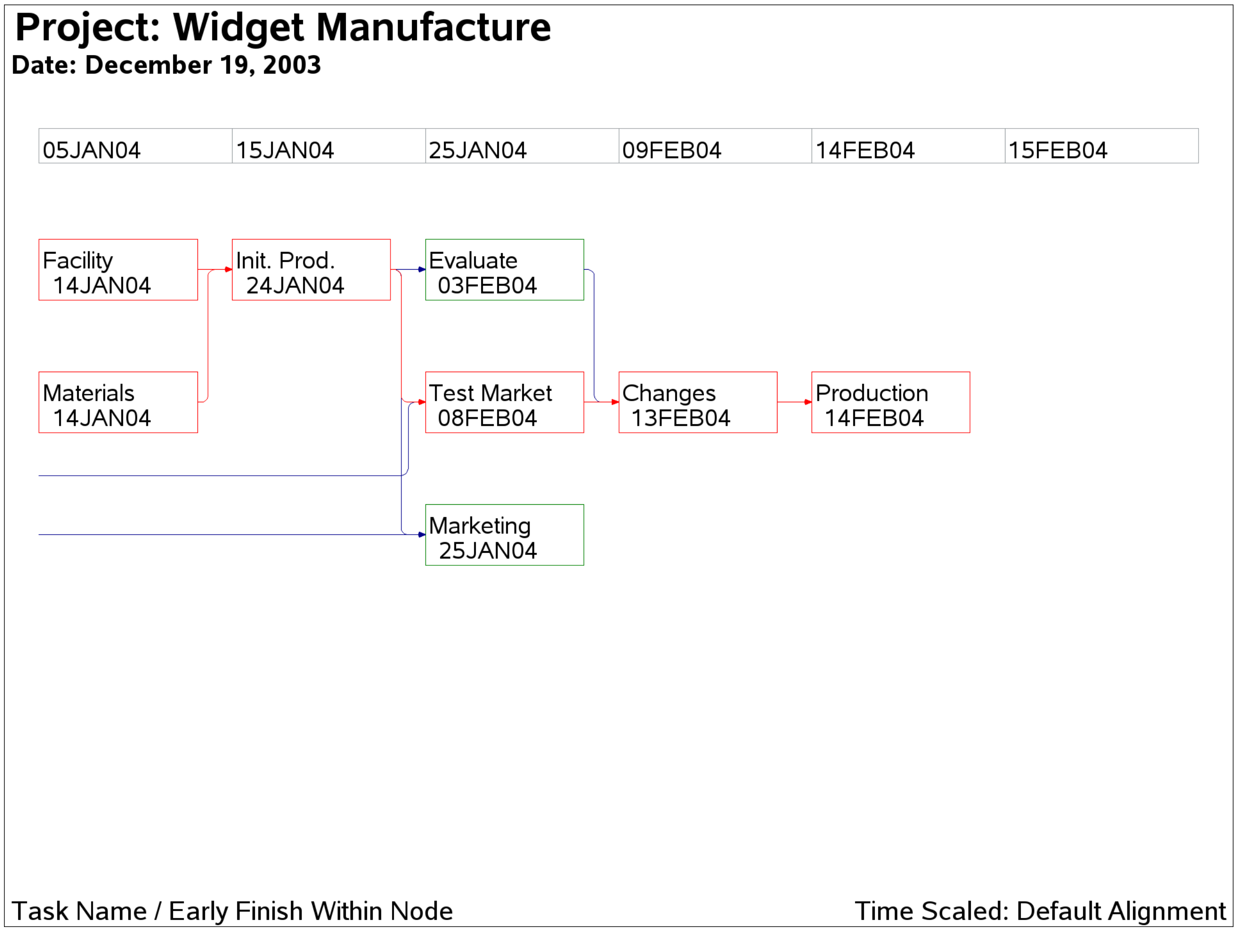
```
pattern1 v=e c=green;
pattern2 v=e c=red;

title  j=l h=3 ' Project: Widget Manufacture';
title2 j=l h=2 ' Date: December 19, 2003';
footnote j=l h=2 ' Task Name / Early Finish Within Node'
         j=r h=2 'Time Scaled: Default Alignment ';
proc netdraw data=widgupd graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            ybetween = 8
```

```
              separatearcs
              novcenter
              id=(task e_finish) nodefid
              nolabel
              showstatus
              carcs=darkblue
              ccritarcs=red
              vmargin=5
              hmargin=5
              timescale
              htext=2  pcompress
              hpages=2 vpages=1
              nopagenumber;
     run;
```

Output 9.9.1 TIMESCALE Option: Default Alignment



Output 9.9.1 *continued*



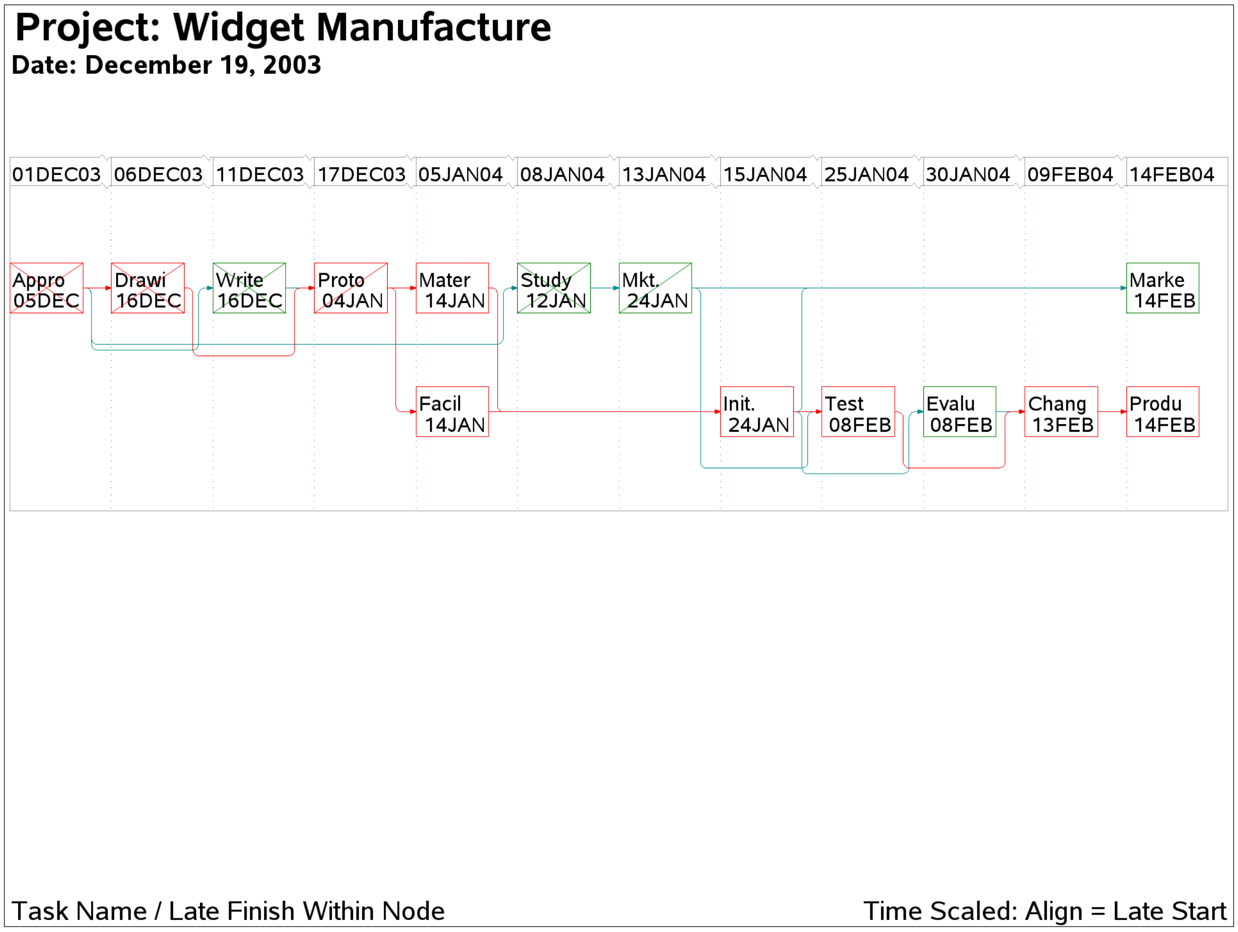
Next, PROC NETDRAW is invoked with several of the time-scale options:

- The ALIGN= option requests that the activities be placed according to the L_START times.
- The FRAME option produces a border around the network diagram.
- The AUTOREF option draws reference lines at every tick mark.
- The LREF= and CREF= options specify the line style and color for the reference lines.
- The SHOWBREAK option requests that breaks in the time axis be indicated by breaks before the corresponding tick marks.

```
footnote j=l h=2 ' Task Name / Late Finish Within Node'
         j=r h=2 'Time Scaled: Align = Late Start ';
proc netdraw data=widgupd graphics;
   actnet / act=task
            succ=(succ1 succ2 succ3)
            ybetween = 10
            separatearcs
```

```
                  pcompress
                  novcenter
                  id=(task l_finish) nodefid
                  nolabel
                  boxwidth=5
                  showstatus
                  carcs=darkcyan
                  ccritarcs=red
                  vmargin=10
                  align=l_start
                  frame
                  autoref
                  lref=33
                  cref=darkcyan
                  showbreak
                  htext=2;
         run;
```

Output 9.9.2 Timescale Option: ALIGN= L_START



Example 9.10: Further Time-Scale Options

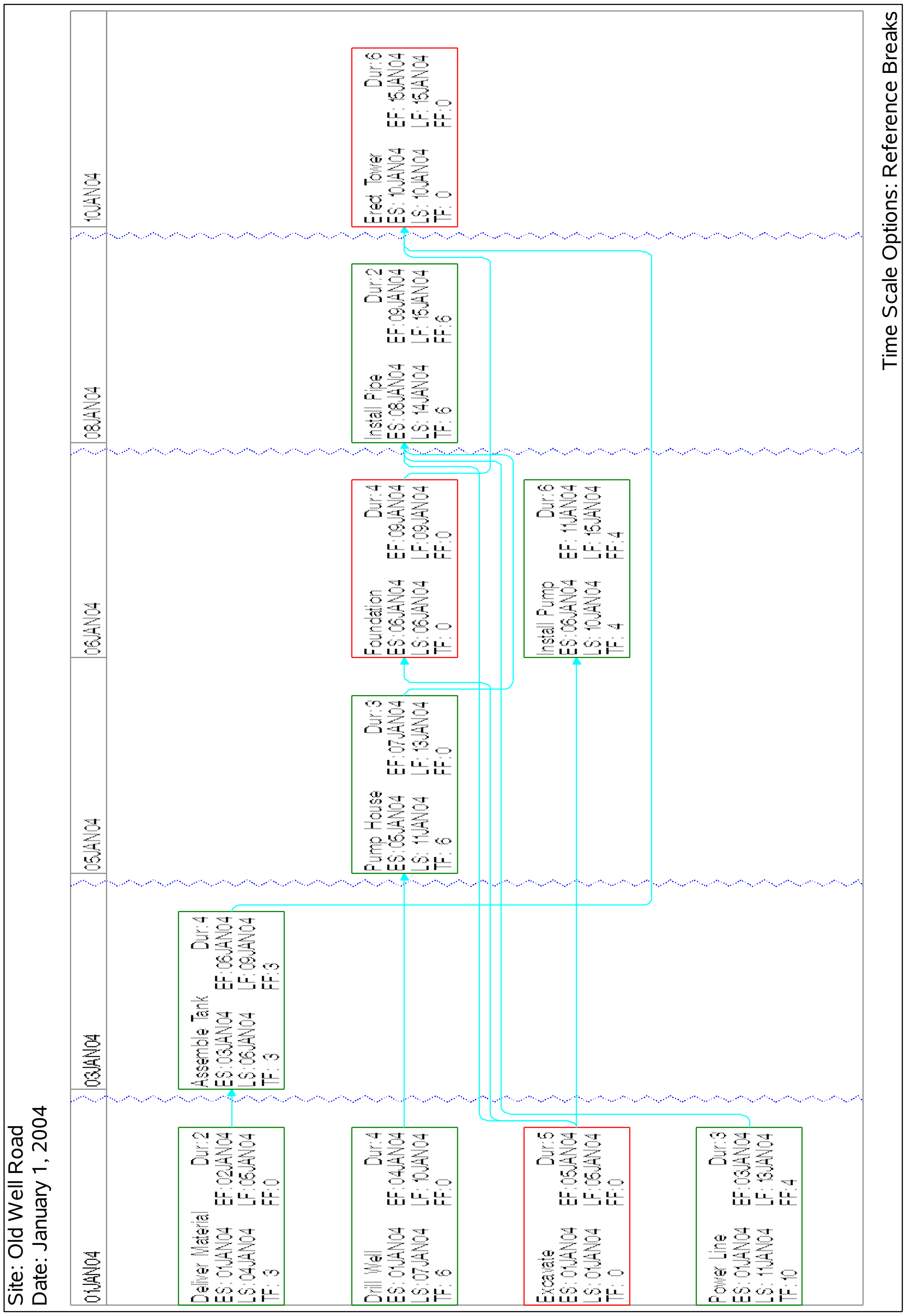
In this example, the construction project described in Example 9.7 is used to illustrate some more time-scale options. First, the REFBREAK option indicates breaks in the time axis by drawing a zigzag line down the diagram before each tick mark corresponding to a break. The CREFBRK= and LREFBRK= options control the color and line style for these lines. The network diagram is shown in Output 9.10.1.

```
title j=l h=1.5 ' Site: Old Well Road';
title2 j=l h=1.5 ' Date: January 1, 2004';
footnote j=r h=1.5 'Time Scale Options: Reference Breaks ';


proc netdraw data=sched graphics;
   actnet / act = task
            dur = duration
            succ = (succesr1-succesr3)
            dp compress separatearcs
            font=swiss htext=2
            timescale refbreak lrefbrk = 33
            carcs=cyan crefbrk = blue;
   run;
```

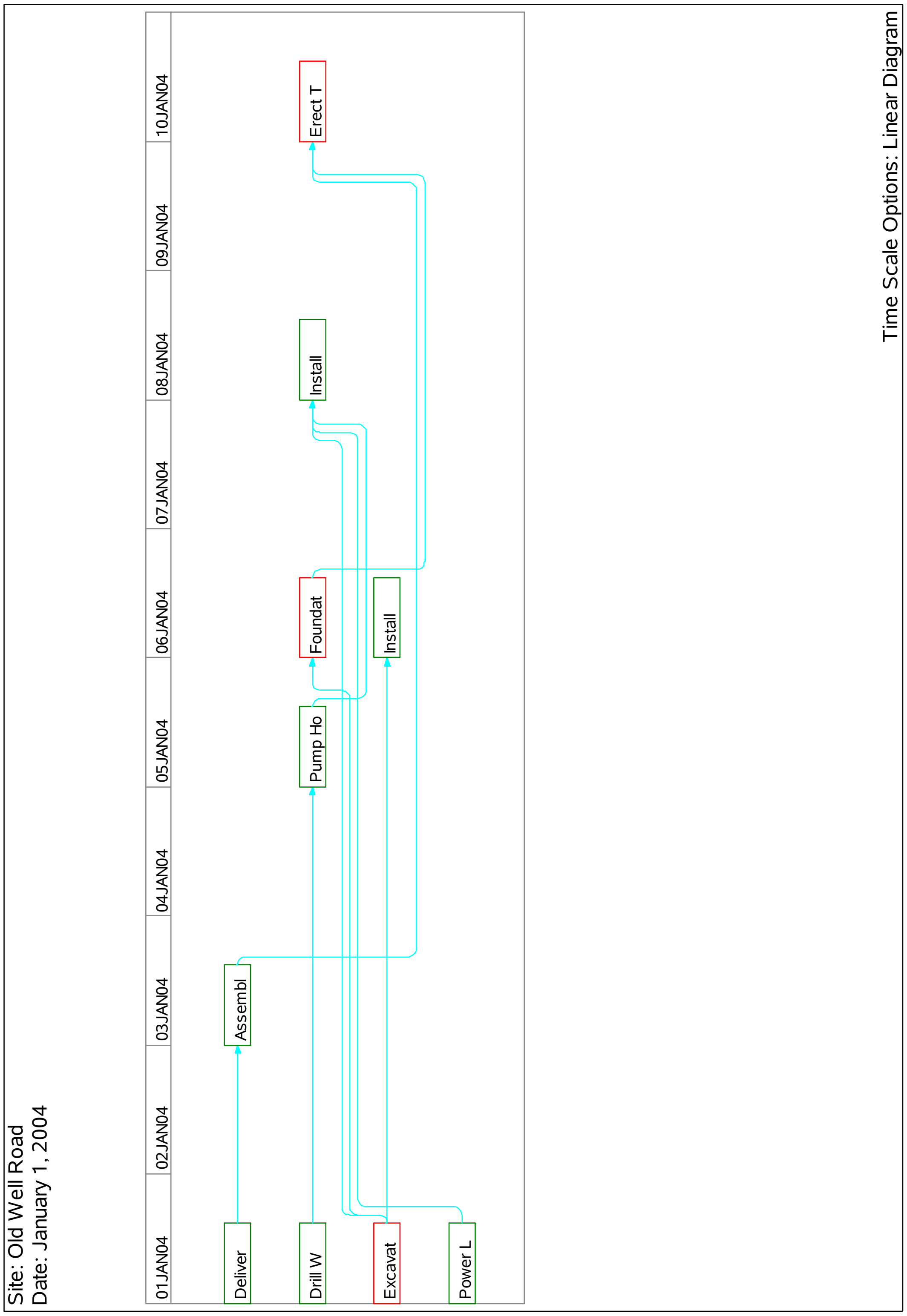
Next, PROC NETDRAW is invoked with the LINEAR option so that there is no break in the time axis. The BOXWIDTH= option limits the size of each node. The diagram is drawn in Output 9.10.2.

```
title j=l h=1.5 ' Site: Old Well Road';
title2 j=l h=1.5 ' Date: January 1, 2004';
footnote j=r h=1.5 'Time Scale Options: Linear Diagram ';

proc netdraw data=sched graphics;
   actnet / act = task
            dur = duration
            succ = (succesr1-succesr3)
            dp
            pcompress
            novcenter
            vmargin = 10
            separatearcs
            htext=2
            carcs=cyan
            id=(task)
            nodefid
            nolabel
            boxwidth=7
            timescale
            linear
            frame;
   run;
```

Output 9.10.1 Time-Scaled Network: Reference Breaks


Site: Old Well Road
Date: January 1, 2004
Time Scale Options: Reference Breaks
01JAN04
03JAN04
05JAN04
06JAN04
08JAN04
10JAN04
Deliver Material Dur:2 ES: 01JAN04 EF: 02JAN04 LS: 04JAN04 LF: 05JAN04 TF: 3 FF: 0
Drill Well Dur:4 ES: 01JAN04 EF: 04JAN04 LS: 07JAN04 LF: 10JAN04 TF: 6 FF: 0
Excavate Dur:5 ES: 01JAN04 EF: 05JAN04 LS: 01JAN04 LF: 05JAN04 TF: 0 FF: 0
Power Line Dur:3 ES: 01JAN04 EF: 03JAN04 LS: 11JAN04 LF: 13JAN04 TF: 10 FF: 4
Assemble Tank Dur:4 ES: 03JAN04 EF: 06JAN04 LS: 06JAN04 LF: 09JAN04 TF: 3 FF: 3
Pump House Dur:3 ES: 05JAN04 EF: 07JAN04 LS: 11JAN04 LF: 13JAN04 TF: 6 FF: 0
Foundation Dur:4 ES: 06JAN04 EF: 09JAN04 LS: 06JAN04 LF: 09JAN04 TF: 0 FF: 0
Install Pump Dur:6 ES: 06JAN04 EF: 11JAN04 LS: 10JAN04 LF: 15JAN04 TF: 4 FF: 4
Install Pipe Dur:2 ES: 08JAN04 EF: 09JAN04 LS: 14JAN04 LF: 15JAN04 TF: 6 FF: 6
Erect Tower Dur:6 ES: 10JAN04 EF: 15JAN04 LS: 10JAN04 LF: 15JAN04 TF: 0 FF: 0

Output 9.10.2 Time-Scaled Network: LINEAR Option



Example 9.11: Zoned Network Diagram

This example illustrates zoned network diagrams. The Widget Manufacturing project is used to illustrate some aspects of this feature. The data set DETAILS contains a variable phase, which identifies the phase of each activity in the project. This data set is merged with the Activity data set from Example 9.1, WIDGET, to produce the data set NETWORK that is input to PROC NETDRAW. The ZONE= option divides the network diagram into horizontal zones based on the project phase. The ZONEPAT option causes the activities in each zone to be drawn using a different pattern. The resulting network diagram is shown in Output 9.11.1.

```
data details;
   format task $12. phase $13. descrpt $30. ;
   input task & phase $ descrpt & ;
   datalines;
Approve Plan  Planning       Develop Concept
Drawings      Engineering    Prepare Drawings
Study Market  Marketing      Analyze Potential Markets
Write Specs   Engineering    Write Specifications
Prototype     Engineering    Build Prototype
Mkt. Strat.   Marketing      Develop Marketing Concept
Materials     Manufacturing  Procure Raw Materials
Facility      Manufacturing  Prepare Manufacturing Facility
Init. Prod.   Manufacturing  Initial Production Run
Evaluate      Testing        Evaluate Product In-House
Test Market   Testing        Test Product in Sample Market
Changes       Engineering    Engineering Changes
Production    Manufacturing  Begin Full Scale Production
Marketing     Marketing      Begin Full Scale Marketing
;
data network;
   merge widget details;
   run;

pattern1 v=e c=green;
pattern2 v=e c=red;
pattern3 v=e c=magenta;
pattern4 v=e c=blue;
pattern5 v=e c=cyan;

title  j=l h=1.5 ' Project: Widget Manufacture';
title2 j=l h=1.5 ' Date: December 1, 2003';
footnote j=r h=1.5 'Zoned Network Diagram ';


proc netdraw data=network graphics;
   actnet / act=task succ=(succ1 succ2 succ3)
            separatearcs
            zone=phase
            zonepat
            pcompress
            htext=2;
   label phase = 'Department';
   run;
```

Next, the project is scheduled with PROC CPM, and PROC NETDRAW is invoked with the ZONE= and TIMESCALE options. The nodes are placed in different zones as dictated by the ZONE variable, phase, and are aligned along the time axis as dictated by the default ALIGN variable, E_START. The MININTERVAL= option produces one tick mark per week for the duration of the project. The LREF= option identifies the linestyle of the reference lines and the dividing lines between zones. The nodes are colored red or green according to whether or not the corresponding activities are critical (PATTERN statements 1 and 2 from the previous invocation of PROC NETDRAW are still valid).

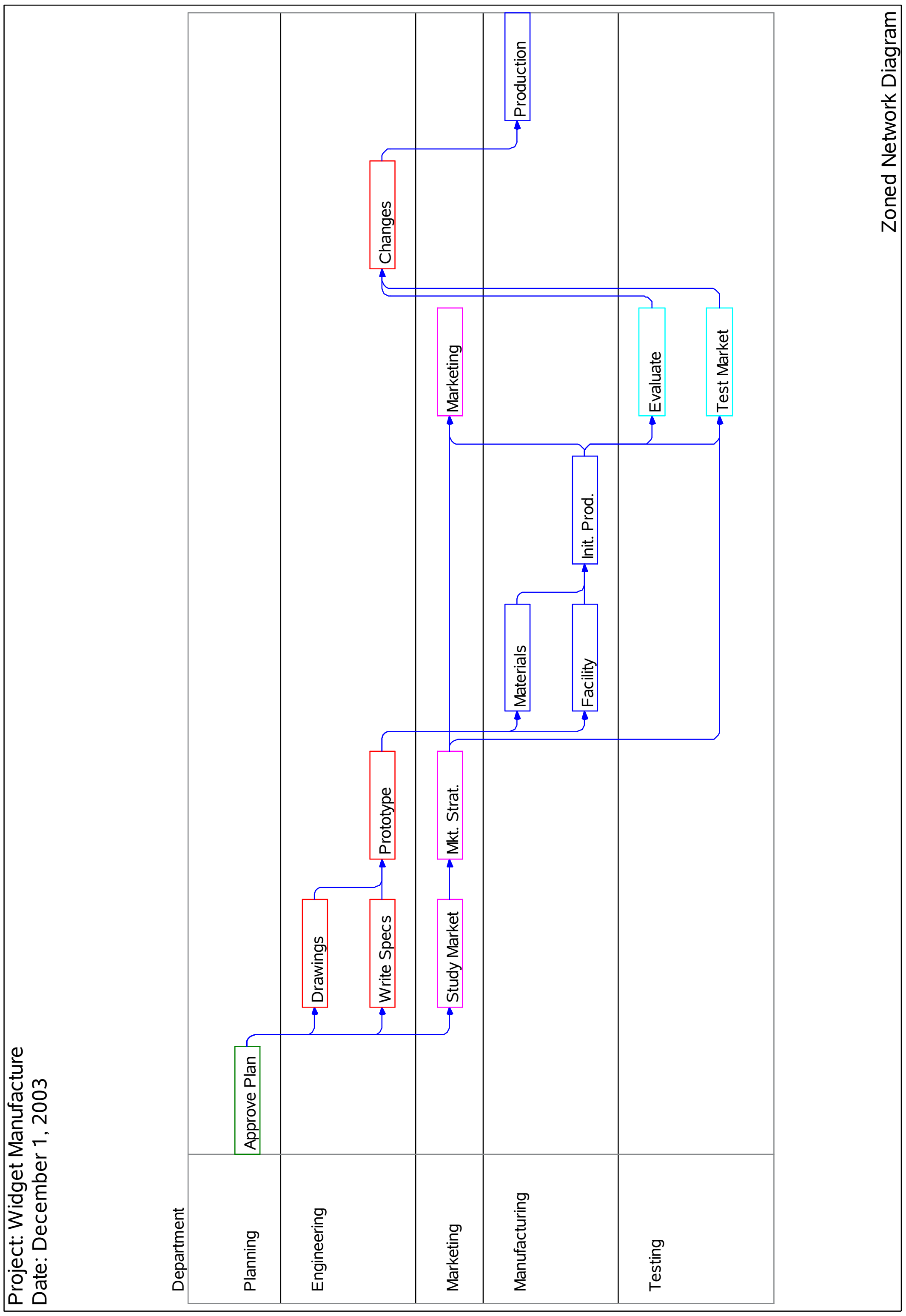
```
proc cpm data=network interval=weekday
         out=sched date='1dec03'd;
   activity task;
   succ     succ1 succ2 succ3;
   duration days;
   id phase;
   run;

title  j=l h=1.5 ' Project: Widget Manufacture';
title2 j=l h=1.5 ' Date: December 1, 2003';
footnote j=r h=1.5 'Zone and Timescale ';
proc netdraw data=sched graphics;
   actnet / act=task succ=(succ1 succ2 succ3)
            pcompress
            carcs = blue ccritarcs=red
            cref = cyan
            caxis = magenta
            lref = 33
            id = (task)
            nodefid
            nolabel
            boxwidth = 8
            htext=2
            separatearcs
            timescale
            mininterval=week
            autoref
            linear
            zone=phase
            zonespace;
   label phase = 'Department';
   run;
```

Example 9.12: Schematic Diagrams

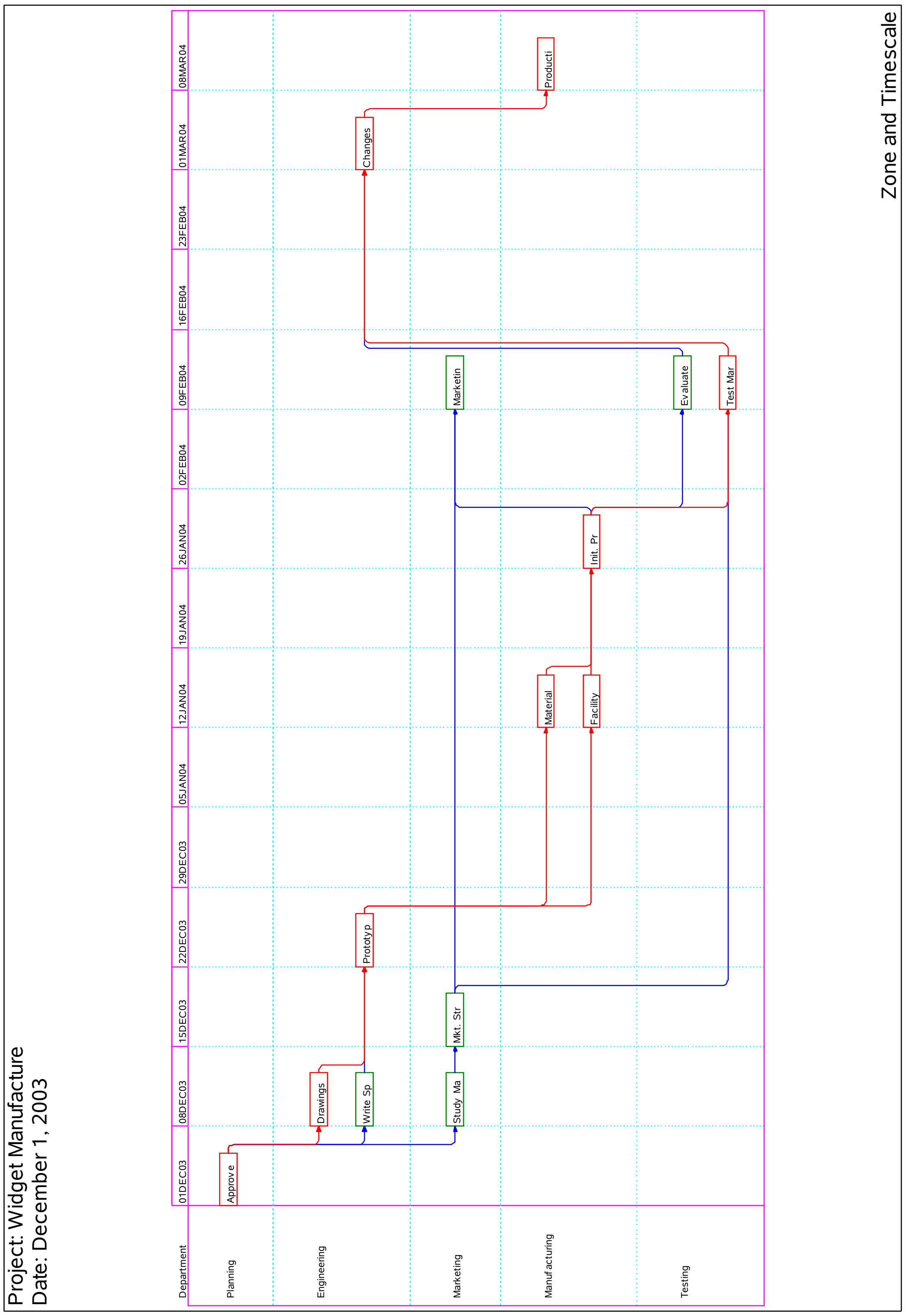
You can use PROC NETDRAW to determine node placement and arc routing for any network depicting a set of nodes connected by arcs. If you want the procedure to determine the node placement, the network must be acyclic. This example illustrates the use of PROC NETDRAW to draw two networks that represent different schematic flows. The first network does not contain any cycles, while the second one has one cycle; to draw the second network, you need to use the BREAKCYCLE option.

First, a schematic representation of the data flow going in and out of the three procedures (CPM, GANTT, and NETDRAW) is drawn using PROC NETDRAW. (See Chapter 3, “Introduction to Project Management,”

Output 9.11.1 Zoned Network Diagram



Output 9.11.2 Zoned Network Diagram with Time Axis

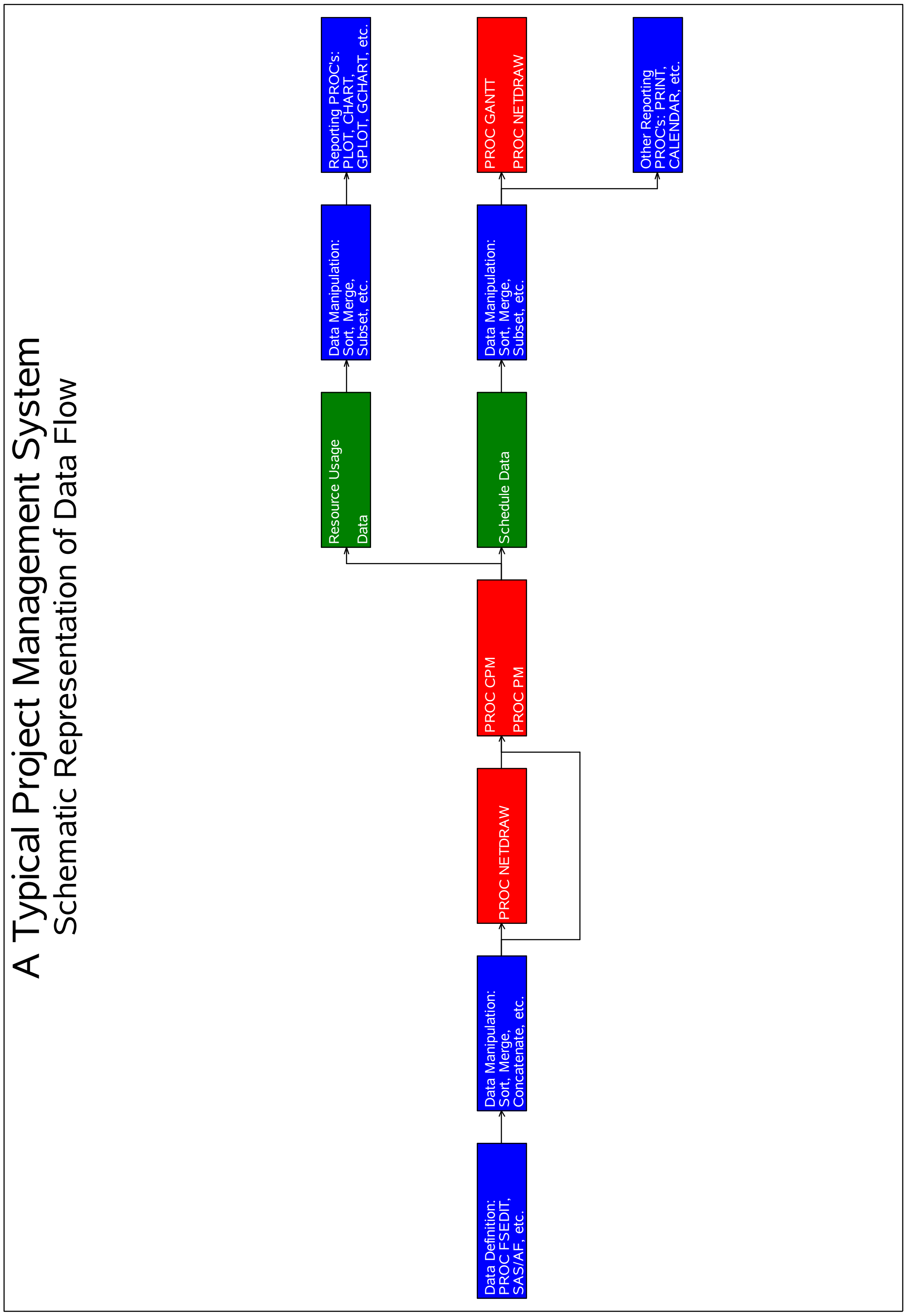


for a detailed discussion of such a data flow.) The PATTERN= option is used to specify the variable in the data set that identifies the color that is to be used for each node. Nodes representing SAS/OR procedures are colored red, the ones representing output data sets are colored green, and all other nodes (representing the use of other parts of the SAS System) are colored blue. Three ID variables are used to specify the text that is to be placed within each node. The flow diagram is shown in Output 9.12.1.

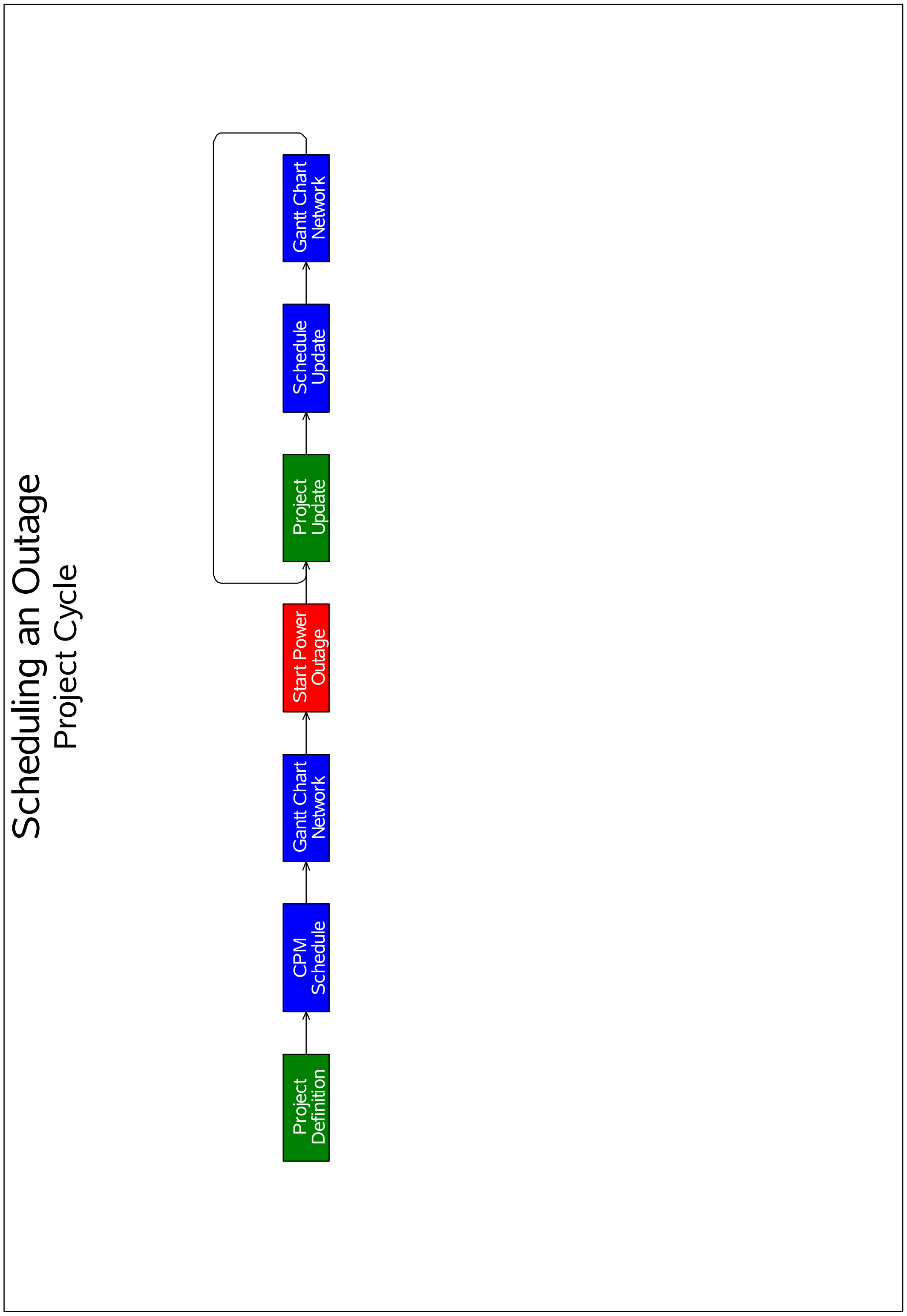
```
data dataflow;
   format id1 $18. id2 $14. id3 $19. ;
   input a $ b $ id1 & id2 & id3 & style;
   datalines;
A B Data Definition:    PROC FSEDIT,    SAS/AF, etc.         2
B C Data Manipulation:  Sort, Merge,    Concatenate, etc.    2
B D Data Manipulation:  Sort, Merge,    Concatenate, etc.    2
D C .                   PROC NETDRAW    .                    1
C E PROC CPM            .               PROC PM              1
C F PROC CPM            .               PROC PM              1
E H Resource Usage      .               Data                 3
F G .                   Schedule Data   .                    3
G I Data Manipulation:  Sort, Merge,    Subset, etc.         2
G J Data Manipulation:  Sort, Merge,    Subset, etc.         2
H K Data Manipulation:  Sort, Merge,    Subset, etc.         2
I . Other Reporting     PROC's: PRINT,  CALENDAR, etc.       2
J . PROC GANTT          .               PROC NETDRAW         1
K . Reporting PROC's:   PLOT, CHART,    GPLOT, GCHART, etc.  2
;

pattern1 v=s c=red;
pattern2 v=s c=blue;
pattern3 v=s c=green;

goptions hpos=110 vpos=70;
title h=3 'A Typical Project Management System';
title2 h=2.5  'Schematic Representation of Data Flow';
proc netdraw data=dataflow graphics;
   actnet / act=a succ=b id = (id1-id3)
            nodefaultid
            nolabel
            pattern=style
            carcs=black coutline=black ctext=white
            hmargin = 2
            ybetween = 15
            rectilinear
            noarrowfill
            pcompress htext=2;
   run;
```

Next, a typical sequence of procedures followed at the scheduling of a nuclear power plant outage is shown using the NETDRAW procedure. Such a schematic diagram is illustrated in Chapter 3, “Introduction to Project Management.” In Figure 3.6, there is a cycle that is not normally allowed in a Network data set that is input to PROC NETDRAW. However, you can draw such networks by specifying the BREAKCYCLE option. (Note that you can also draw cyclic networks by specifying explicitly the node coordinates or an ALIGN= variable that fixes the *x* coordinates for each node.)

Output 9.12.1 Schematic Representation of Data Flow


A Typical Project Management System
Schematic Representation of Data Flow
Data Definition: PROC FSEDIT, SAS/AF, etc.
Data Manipulation: Sort, Merge, Concatenate, etc.
PROC NETDRAW
PROC CPM PROC PM
Resource Usage Data
Schedule Data
Data Manipulation: Sort, Merge, Subset, etc.
Data Manipulation: Sort, Merge, Subset, etc.
Reporting PROC's: PLOT, CHART, GPLOT, GCHART, etc.
PROC GANTT PROC NETDRAW
Other Reporting PROC's: PRINT, CALENDAR, etc.

Output 9.12.2 Scheduling a Power Outage



In this example, the data set OUTAGE contains the network representation. The variable style is used to color nodes appropriately. The resulting diagram is shown in Output 9.12.2.

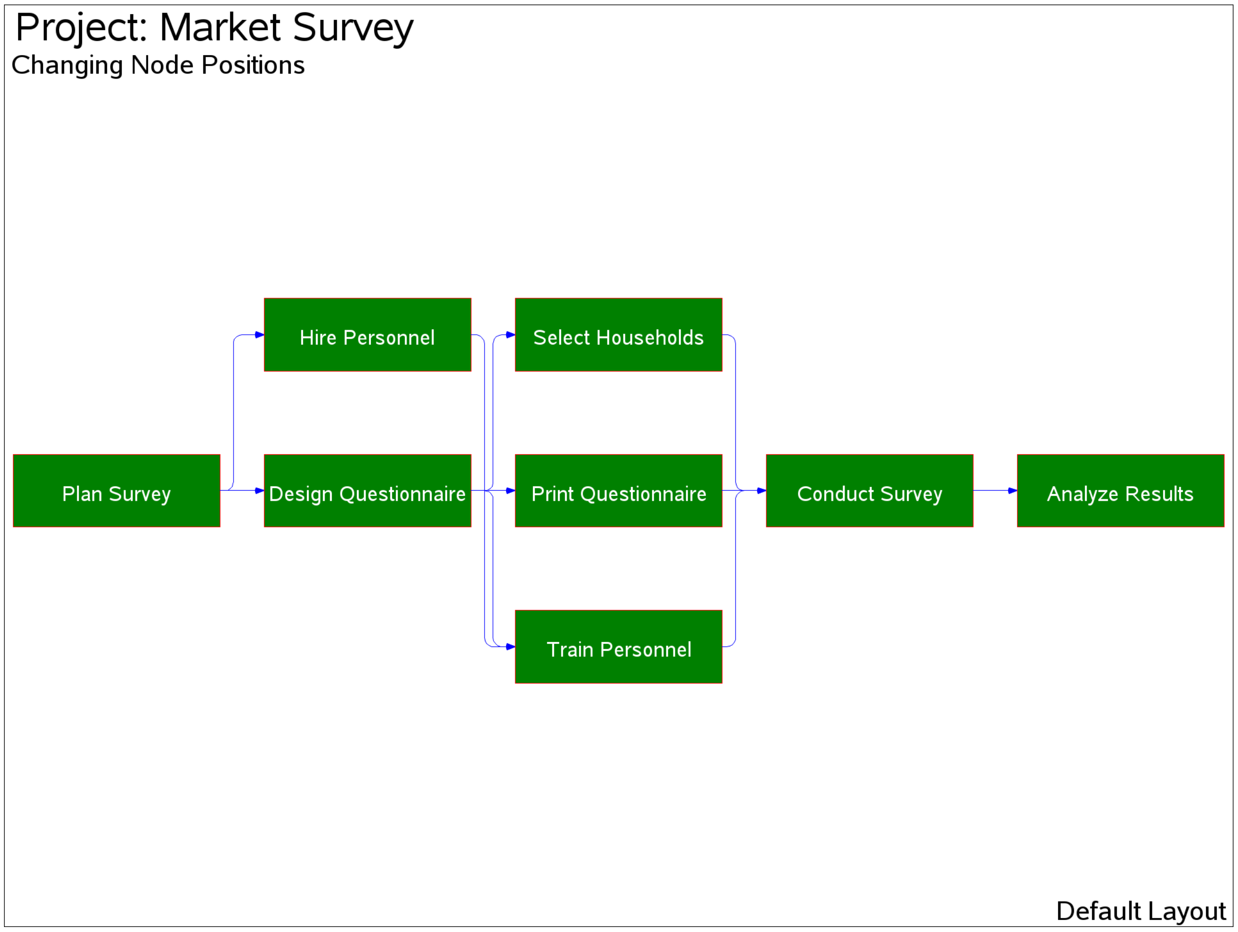
```
data outage;
   input a $ b $ id1 $20.  id2 $20. style;
   datalines;
A   B   Project           Definition            1
B   C   CPM               Schedule              2
C   D   Gantt Chart       Network               3
D   E   Start Power       Outage                4
E   F   Project           Update                1
F   G   Schedule          Update                2
G   E   Gantt Chart       Network               3
;

goptions hpos=110 vpos=70;
title h=3 'Scheduling an Outage';
title2 h=2.5  'Project Cycle';

pattern1 v=s c=green;
pattern2 v=s c=blue;
pattern3 v=s c=blue;
pattern4 v=s c=red;

proc netdraw data=outage graphics;
   actnet / act=a succ=b id = (id1 id2)
            breakcycle
            nodefaultid centerid
            vmargin = 5 hmargin = 0
            nolabel novcenter
            pattern=style
            carcs=black coutline=black ctext=white
            ybetween = 15 xbetween=3
            noarrowfill
            pcompress htext=2;
   run;
```

Example 9.13: Modifying Network Layout

This example uses the SURVEY project described in Chapter 3, “Introduction to Project Management,” to illustrate how you can modify the default layout of the network. The data set SURVEY contains the project information. PROC NETDRAW is invoked with the GRAPHICS option. The network diagram is shown in Output 9.13.1.

```
data survey;
   format id $20. activity succ1-succ3 $8. phase $9. ;
   input id        &
         activity  &
         duration
         succ1     &
         succ2     &
         succ3     &
         phase     $ ;
   datalines;
Plan Survey           plan sur   4 hire per  design q  .        Plan
Hire Personnel        hire per   5 trn per   .         .        Prepare
Design Questionnaire  design q   3 trn per   select h  print q  Plan
Train Personnel       trn per    3 cond sur  .         .        Prepare
Select Households     select h   3 cond sur  .         .        Prepare
Print Questionnaire   print q    4 cond sur  .         .        Prepare
Conduct Survey        cond sur  10 analyze   .         .        Implement
Analyze Results       analyze    6 .         .         .        Implement
;


pattern1 v=s c=green;

title j=l h=3' Project: Market Survey';
title2 j=l h=2 ' Changing Node Positions';
footnote j=r h=2 'Default Layout ';

proc netdraw data=survey graphics out=network;
   actnet / act=activity
            succ=(succ1-succ3)
            id=(id) nodefid nolabel
            carcs = blue
            ctext  = white
            coutline=red
            centerid
            boxht = 3
            htext=2
            pcompress
            separatearcs
            ybetween=8;
   run;

footnote;
title2 'NETWORK Output Data Set';
proc print data=network;
   run;
```

Output 9.13.1 Default Network Layout of SURVEY Project



The Layout data set produced by PROC NETDRAW (displayed in Output 9.13.2) contains the *x* and *y* coordinates for all the nodes in the network and for all the turning points of the arcs connecting them.

Suppose that you want to interchange the positions of the nodes corresponding to the two activities, 'Select Households' and 'Train Personnel.' As explained in the section "Controlling the Layout" on page 707, you can invoke the procedure in FULLSCREEN mode and use the MOVE command to move the nodes to desired locations. In this example, the data set NETWORK produced by PROC NETDRAW is used to change the *x* and *y* coordinates of the nodes. A new data set called NODEPOS is created from NETWORK by retaining only the observations containing node positions (recall that for such observations, _SEQ_ = '0') and by dropping the _SEQ_ variable. Further, the *y* coordinates (given by the values of the _Y_ variable) for the two activities 'Select Households' and 'Train Personnel' are interchanged. The new data set, displayed in Output 9.13.3, is then input to PROC NETDRAW.

Output 9.13.2 Layout Data Set

Project: Market Survey

NETWORK Output Data Set

Obs	_FROM_	_TO_	_X_	_Y_	_SEQ_	_PATTERN	id
1	plan sur	hire per	1.0	2	0	1	Plan Survey
2	plan sur	hire per	1.5	2	1	.	Plan Survey
3	plan sur	hire per	1.5	3	2	.	Plan Survey
4	plan sur	design q	1.0	2	0	1	Plan Survey
5	hire per	trn per	2.0	3	0	1	Hire Personnel
6	hire per	trn per	2.5	3	1	.	Hire Personnel
7	hire per	trn per	2.5	1	2	.	Hire Personnel
8	design q	trn per	2.0	2	0	1	Design Questionnaire
9	design q	trn per	2.5	2	1	.	Design Questionnaire
10	design q	trn per	2.5	1	2	.	Design Questionnaire
11	design q	select h	2.0	2	0	1	Design Questionnaire
12	design q	select h	2.5	2	1	.	Design Questionnaire
13	design q	select h	2.5	3	2	.	Design Questionnaire
14	design q	print q	2.0	2	0	1	Design Questionnaire
15	trn per	cond sur	3.0	1	0	1	Train Personnel
16	trn per	cond sur	3.5	1	1	.	Train Personnel
17	trn per	cond sur	3.5	2	2	.	Train Personnel
18	select h	cond sur	3.0	3	0	1	Select Households
19	select h	cond sur	3.5	3	1	.	Select Households
20	select h	cond sur	3.5	2	2	.	Select Households
21	print q	cond sur	3.0	2	0	1	Print Questionnaire
22	cond sur	analyze	4.0	2	0	1	Conduct Survey
23	analyze		5.0	2	0	1	Analyze Results

```
data nodepos;
   set network;
   if _seq_ = 0;
   drop _seq_;
   if _from_ = 'select h' then _y_=1;
   if _from_ = 'trn per' then _y_=3;
   run;

title2 'Modified Node Positions';
proc print data=nodepos;
   run;
```

Output 9.13.3 Modified Layout Data Set

Project: Market Survey

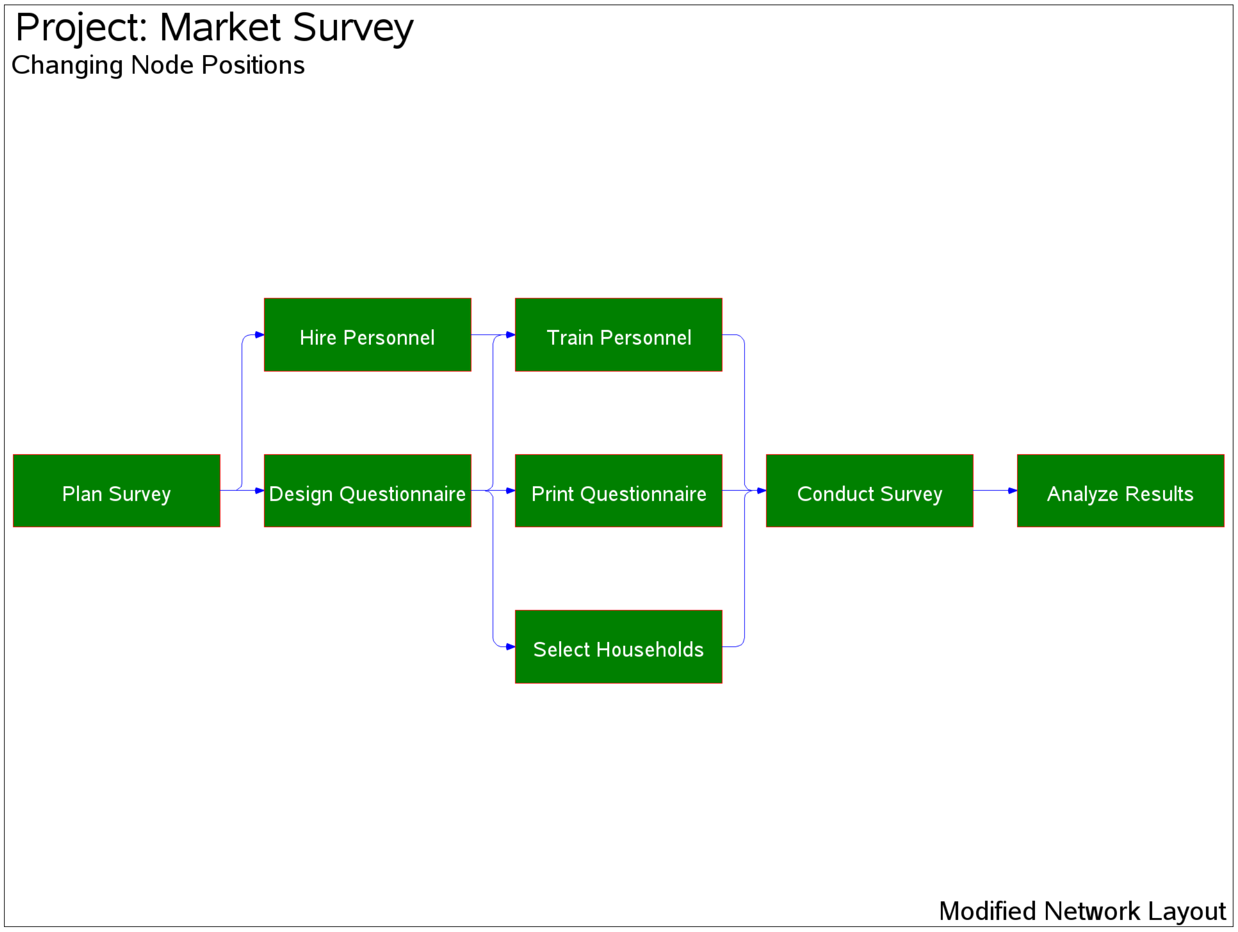
Modified Node Positions

Obs	_FROM_	_TO_	_X_	_Y_	_PATTERN	id
1	plan sur	hire per	1	2	1	Plan Survey
2	plan sur	design q	1	2	1	Plan Survey
3	hire per	trn per	2	3	1	Hire Personnel
4	design q	trn per	2	2	1	Design Questionnaire
5	design q	select h	2	2	1	Design Questionnaire
6	design q	print q	2	2	1	Design Questionnaire
7	trn per	cond sur	3	3	1	Train Personnel
8	select h	cond sur	3	1	1	Select Households
9	print q	cond sur	3	2	1	Print Questionnaire
10	cond sur	analyze	4	2	1	Conduct Survey
11	analyze		5	2	1	Analyze Results

Note that the data set NODEPOS contains variables named _FROM_ and _TO_, which specify the (activity, successor) information; hence, the call to PROC NETDRAW does not contain the ACTIVITY= and SUCCESSOR= specifications. The presence of the variables _X_ and _Y_ indicates to PROC NETDRAW that the data set contains the *x* and *y* coordinates for all the nodes. Because there is no variable named _SEQ_ in this data set, PROC NETDRAW assumes that only the node coordinates are given and uses these node positions to determine how the arcs are to be routed. The resulting network diagram is shown in Output 9.13.4.

```
title j=l h=3 ' Project: Market Survey';
title2 j=l h=2 ' Changing Node Positions';
footnote j=r h=2 'Modified Network Layout ';

proc netdraw data=nodepos graphics;
   actnet / id=(id) nodefid nolabel
            carcs = blue
            ctext = white
            coutline = red
            centerid
            boxht = 3
            htext=2
            pcompress
            separatearcs
            ybetween=8;
   run;
```

Output 9.13.4 Modified Network Layout of SURVEY Project



Example 9.14: Specifying Node Positions

This example uses a typical problem in network flow optimization to illustrate how you can use PROC NETDRAW to draw a network by specifying completely all the node positions. Consider a simple two-period production inventory problem with one manufacturing plant (PLANT), two warehouses (DEPOT1 and DEPOT2), and one customer (CUST). In each period, the customer can receive goods directly from the plant or from the two warehouses. The goods produced at the plant can be used to satisfy directly some or all of the customer's demands or can be shipped to a warehouse. Some of the goods can also be carried over to the next period as inventory at the plant. The problem is to determine the minimum cost of satisfying the customer's demands; in particular, how much of the customer's demands in each period is to be satisfied from the inventory at the two warehouses or from the plant, and also how much of the production is to be carried over as inventory at the plant? This problem can be solved using PROC NETFLOW; the details are not discussed here. Let PLANT_`i` represent the production at the plant in period `i`, DEPOT1_`i` represent the inventory at DEPOT1 in period `i`, DEPOT2_`i` represent the inventory at DEPOT2 in period `i`, and CUST_`i` represent the customer's demand in period `i` (`i` =1, 2). These variables can be thought of as nodes in a network with the following data representing the COST and CAPACITY of the arcs connecting them:

```
FROM      TO          COST    CAPACITY

PLANT_1   CUST_1       10        75
PLANT_1   DEPOT1_1      7        75
PLANT_1   DEPOT2_1      8        75
DEPOT1_1  CUST_1        3        20
DEPOT2_1  CUST_1        2        10
PLANT_1   PLANT_2       2       100
DEPOT1_1  DEPOT1_2      1       100
DEPOT2_1  DEPOT2_2      1       100
PLANT_2   CUST_2       10        75
PLANT_2   DEPOT1_2      7        75
PLANT_2   DEPOT2_2      8        75
DEPOT1_2  CUST_2        3        20
DEPOT2_2  CUST_2        2        10
CUST_1    .             .         .
CUST_2    .             .         .
```

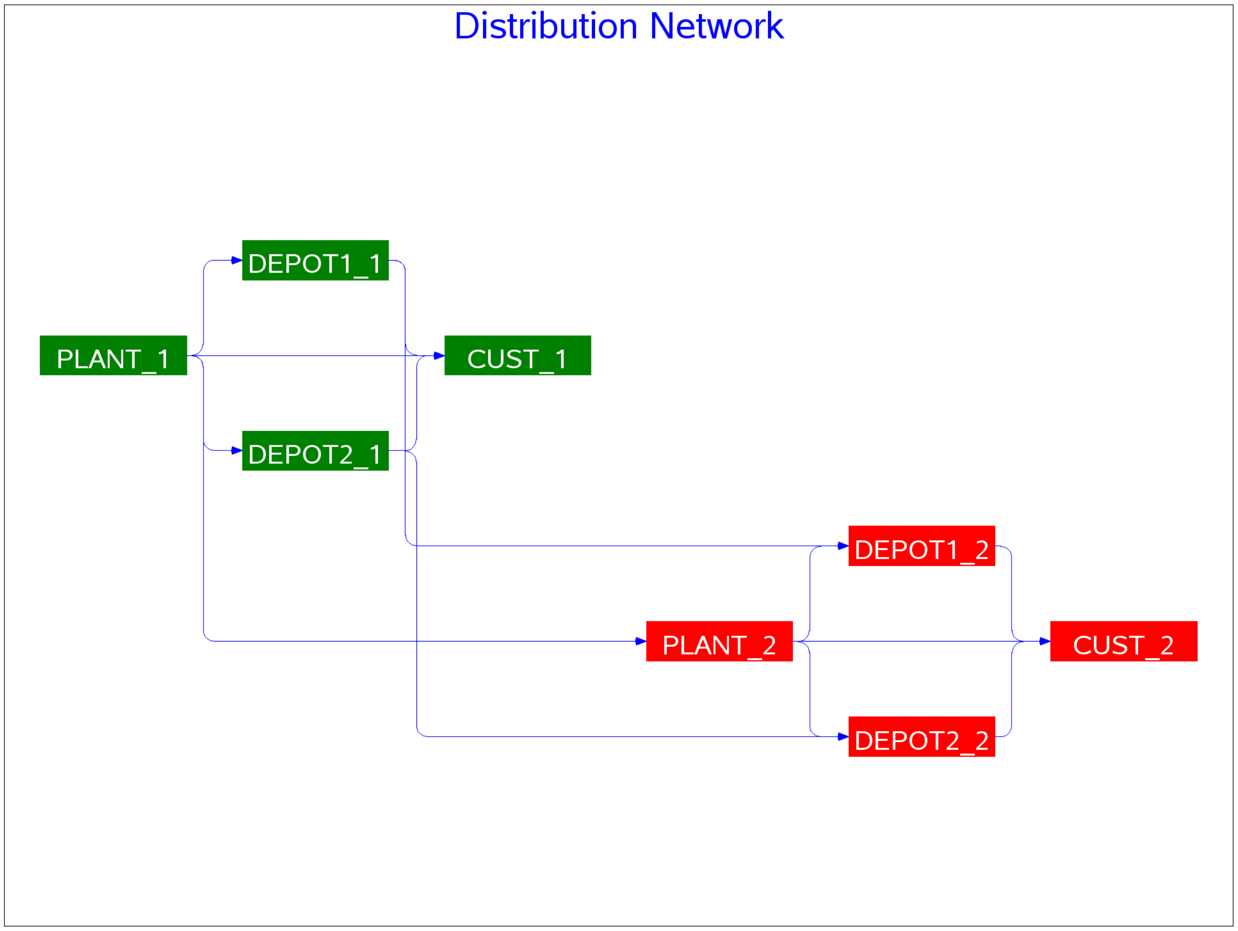
Suppose that you want to use PROC NETDRAW to draw the network corresponding to the preceding network flow problem and suppose also that you require the nodes to be placed in specific positions. The following program saves the network information along with the required node coordinates in the Network data set ARCS and invokes PROC NETDRAW to draw the network diagram shown in Output 9.14.1. The Network data set also contains a variable named _pattern, which specifies that pattern statement 1 be used for nodes relating to period 1 and pattern statement 2 be used for those relating to period 2.

```
data arcs;
   input from $  to $  _x_ _y_ _pattern;
   datalines;
PLANT_1  CUST_1      1  5  1
PLANT_1  DEPOT1_1    1  5  1
PLANT_1  DEPOT2_1    1  5  1
DEPOT1_1 CUST_1      2  6  1
DEPOT2_1 CUST_1      2  4  1
PLANT_1  PLANT_2     1  5  1
DEPOT1_1 DEPOT1_2    2  6  1
DEPOT2_1 DEPOT2_2    2  4  1
PLANT_2  CUST_2      4  2  2
PLANT_2  DEPOT1_2    4  2  2
PLANT_2  DEPOT2_2    4  2  2
DEPOT1_2 CUST_2      5  3  2
DEPOT2_2 CUST_2      5  1  2
CUST_1     .         3  5  1
CUST_2     .         6  2  2
;

title c=blue 'Distribution Network';
pattern1 v=s  c=green;
pattern2 v=s  c=red;
proc netdraw data=arcs graphics out=netout;
   actnet / act=from succ=to separatearcs
            ybetween = 4
            centerid
            ctext = white
```

```
                  carcs=blue
                  htext=2
                  pcompress;
      run;
```

NOTE: This network diagram can also be drawn by using suitably defined ZONE and ALIGN variables.

Output 9.14.1 Distribution Network

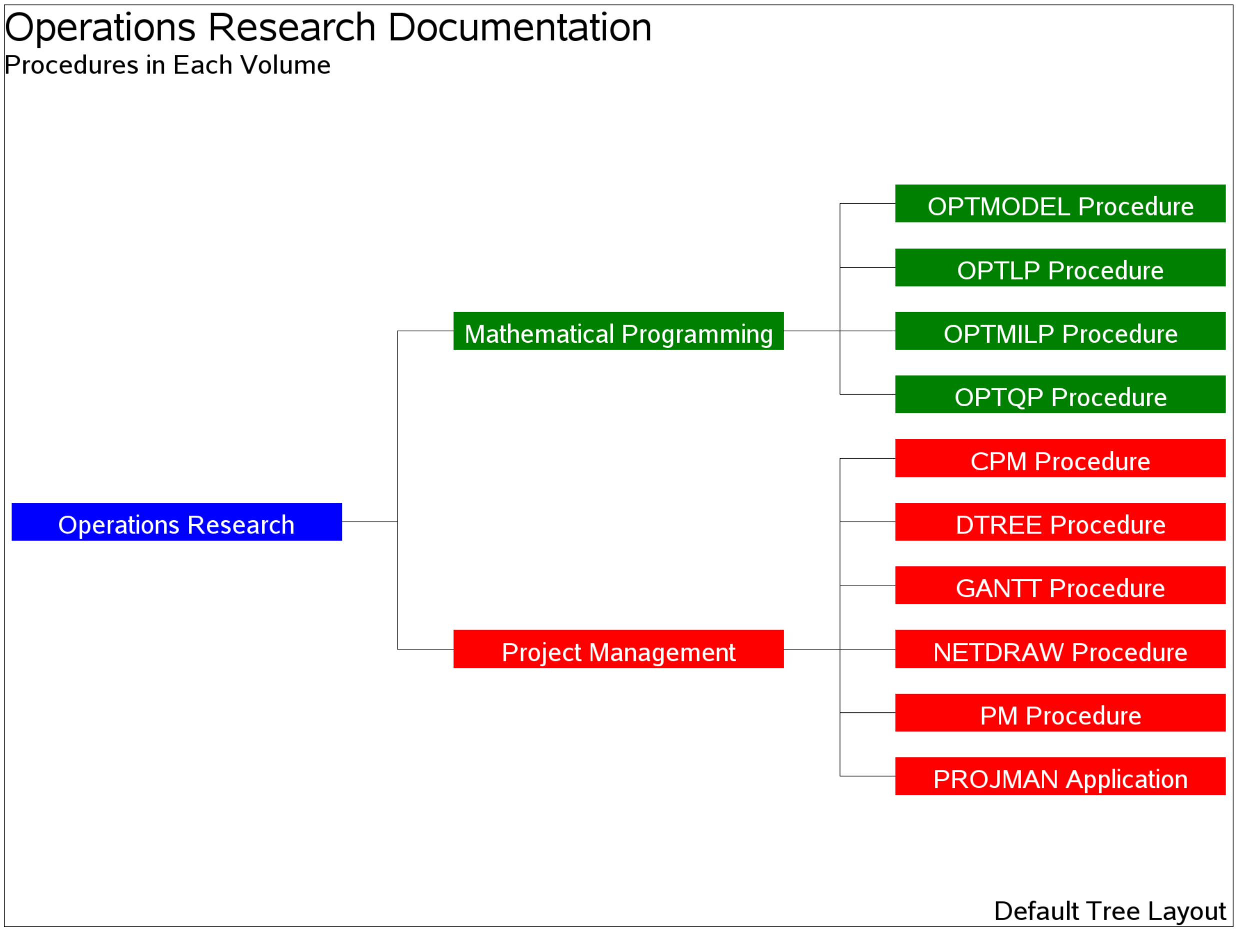


Example 9.15: Organizational Charts with PROC NETDRAW

This example illustrates using the TREE option to draw organizational charts. The Network data set, DOCUMENT, describes how the procedures are distributed between two volumes of the SAS/OR documentation. The structure can be visualized easily in a tree diagram. The data set DOCUMENT contains the parent-child relationship for each node of the diagram. For each node, a detailed description is contained in the variable ID. In addition, the variable _pattern specifies the pattern to be used for each node. PROC NETDRAW is invoked with the TREE option, which illustrates the organization of the documentation in the form of a tree diagram drawn from left to right. The CENTERID option centers text within each node. Arrowheads are not necessary for this diagram and are suppressed by specifying ARROWHEAD=0. Output 9.15.1 shows the resulting diagram.

```
data document;
   format parent child $8. id $24.;
   input parent $ child $ id & _pattern;
   datalines;
OR       MPBOOK     Operations Research       1
OR       PMBOOK     Operations Research       1
PMBOOK   CPM        Project Management        2
PMBOOK   DTREE      Project Management        2
PMBOOK   GANTT      Project Management        2
PMBOOK   NETDRAW    Project Management        2
PMBOOK   PM         Project Management        2
PMBOOK   PROJMAN    Project Management        2
MPBOOK   OPTMODEL   Mathematical Programming  3
MPBOOK   OPTLP      Mathematical Programming  3
MPBOOK   OPTMILP    Mathematical Programming  3
MPBOOK   OPTQP      Mathematical Programming  3
CPM      .          CPM Procedure             2
DTREE    .          DTREE Procedure           2
GANTT    .          GANTT Procedure           2
NETDRAW  .          NETDRAW Procedure         2
PM       .          PM Procedure              2
PROJMAN  .          PROJMAN Application       2
OPTMODEL .          OPTMODEL Procedure        3
OPTLP    .          OPTLP Procedure           3
OPTMILP  .          OPTMILP Procedure         3
OPTQP    .          OPTQP Procedure           3
;

pattern1 v=s c=blue;
pattern2 v=s c=red;
pattern3 v=s c=green;

title j=l h=3 'Operations Research Documentation';
title2 j=l h=2 'Procedures in Each Volume';
footnote j=r h=2 'Default Tree Layout ';
proc netdraw graphics data=document;
   actnet / act=parent
            succ=child
            id=(id)
            nodefid
            nolabel
            pcompress
            centerid
            tree
            xbetween=15
            ybetween=3
            arrowhead=0
            rectilinear
            carcs=black
            ctext=white
            htext=3;
   run;
```

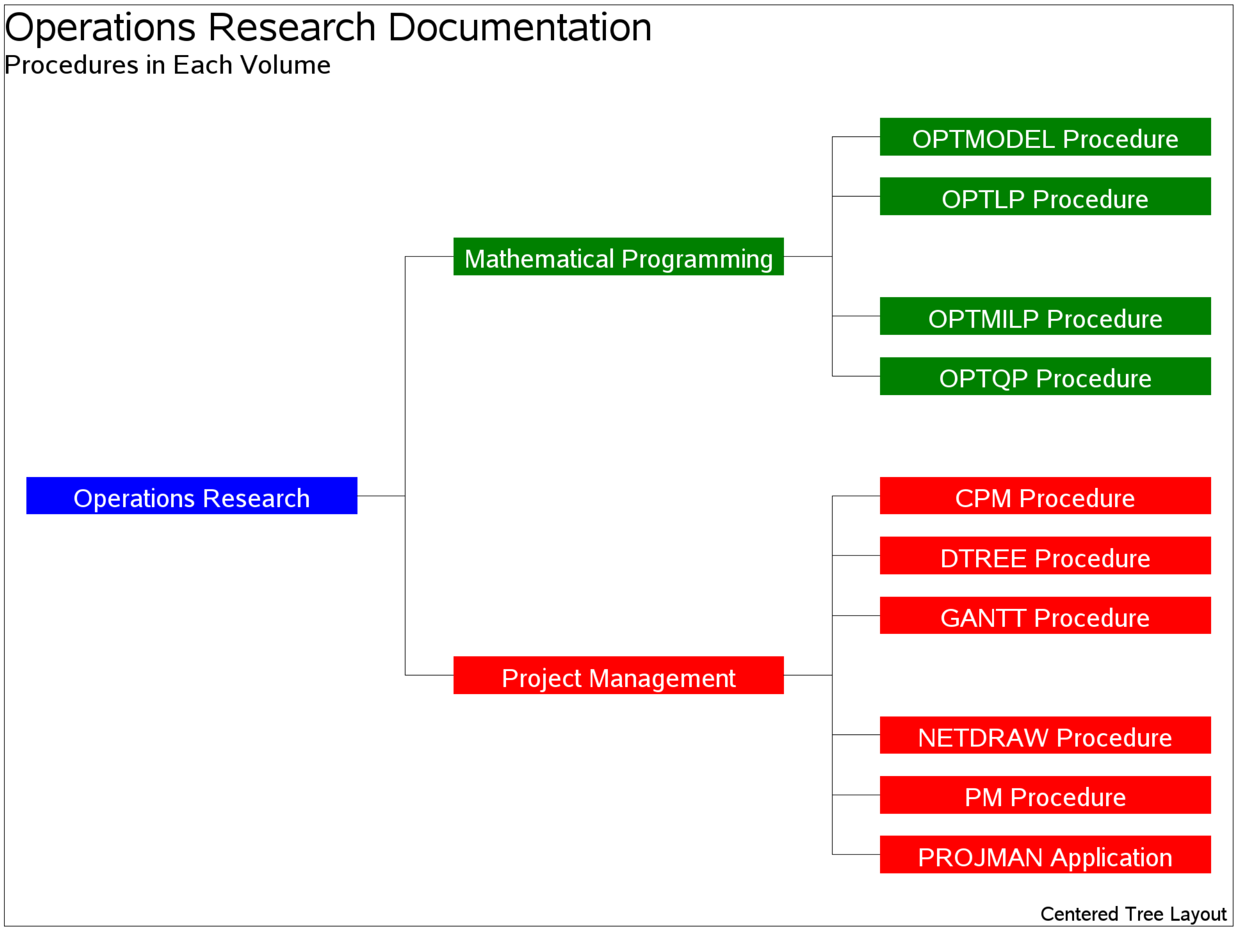
Output 9.15.1 Organization of Documentation: Default TREE Layout



The procedure draws the tree compactly with the successors of each node being placed to the immediate right of the node, ordered from top to bottom in the order of occurrence in the Network data set. The next invocation of PROC NETDRAW illustrates the effect of the SEPARATESONS and CENTERSUBTREE options on the layout of the tree (see Output 9.15.2).

```
footnote j=r h=1.5 'Centered Tree Layout ';
proc netdraw graphics data=document;
   actnet / act=parent
            succ=child
            id=(id)
            nodefid
            nolabel
            pcompress
            novcenter
            centerid
            tree
            separatesons
            centersubtree
            xbetween=15
            ybetween=3
            arrowhead=0
```

```
               rectilinear
               carcs=black
               ctext=white
               htext=3.5;
run;
```



Output 9.15.2 Organization of Documentation: Controlled TREE Layout

Example 9.16: Annotate Facility with PROC NETDRAW

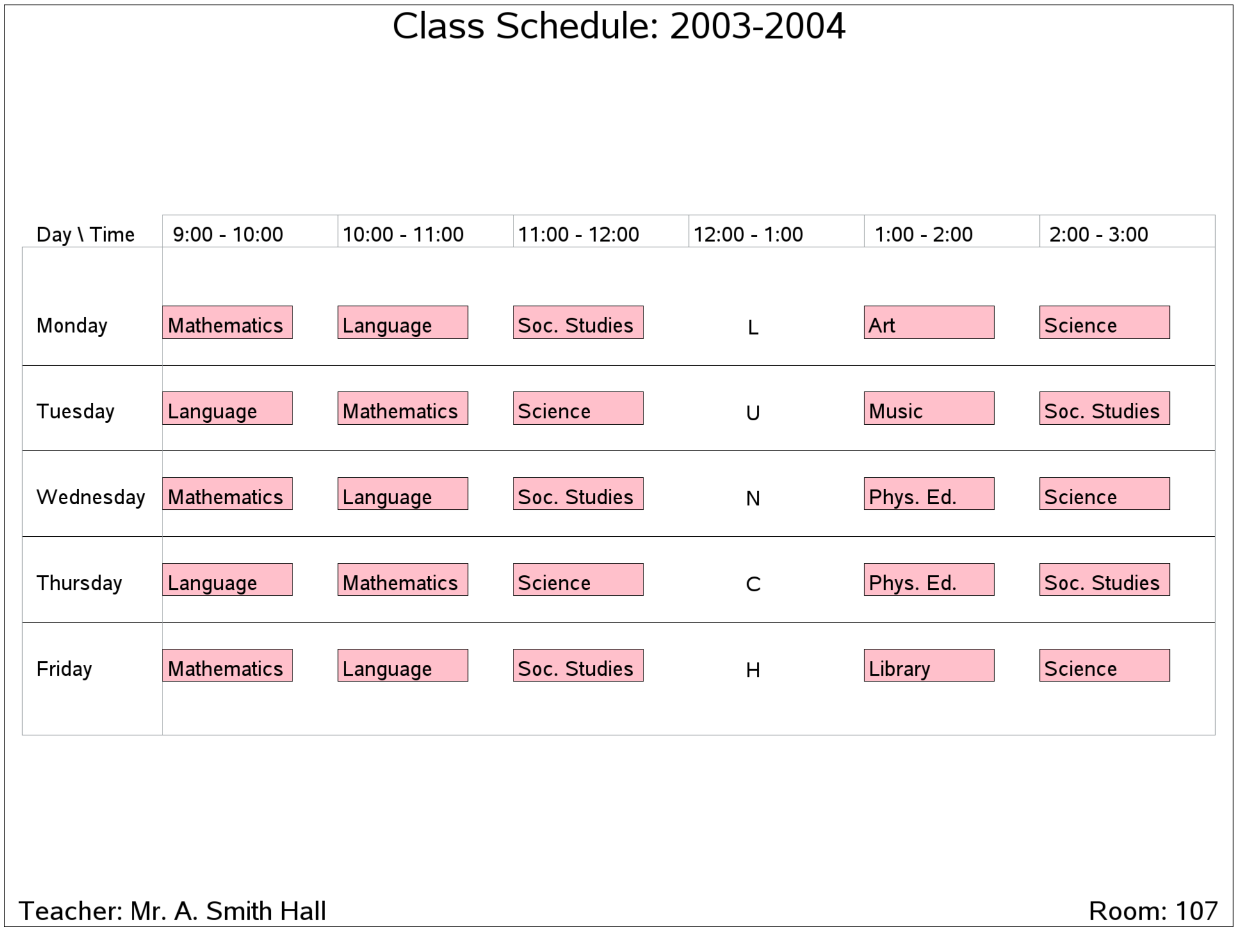
This example demonstrates the use of PROC NETDRAW for a nonstandard application. The procedure is used to draw a time table for a class of students. The days of the week are treated as different zones, and the times within a day are treated as different values of an alignment variable. The following DATA step defines a total of twenty activities, ‘m1’, . . . ,‘f5’, which refer to the five different periods for the five different days of the week. The variable class contains the name of the subject taught in the corresponding period and day. Note that the periods are taught during the hours 1, 2, 3, 5, and 6; the fourth hour is set aside for lunch. The time axis is labeled with the format CLASSTIM, which is defined using PROC FORMAT. The USEFORMAT option in the ACTNET statement instructs PROC NETDRAW to use the explicit format specified for the time variable rather than the default format.

This example also illustrates the use of the Annotate facility with PROC NETDRAW. The data set ANNO labels the fourth period ‘LUNCH.’ The positions for the text are specified using data coordinates that refer to the (_X_, _Y_) grid used by PROC NETDRAW. Thus, for example X = '4' identifies the *x* coordinate for the annotated text to be the fourth period, and the *y* coordinates are set appropriately. The resulting time table is shown in Output 9.16.1.

```
 /* Define format for the ALIGN= variable */
proc format;
   value classtim 1 = ' 9:00 - 10:00'
                  2 = '10:00 - 11:00'
                  3 = '11:00 - 12:00'
                  4 = '12:00 - 1:00 '
                  5 = ' 1:00 - 2:00 '
                  6 = ' 2:00 - 3:00 ';
   run;

data schedule;
   format day $9. class $12. ;
   input day $ class &  time daytime $ msucc $;
   format time classtim.;
   label day = "Day \ Time";
   datalines;
Monday     Mathematics    1   m1   .
Monday     Language       2   m2   .
Monday     Soc. Studies   3   m3   .
Monday     Art            5   m4   .
Monday     Science        6   m5   .
Tuesday    Language       1   t1   .
Tuesday    Mathematics    2   t2   .
Tuesday    Science        3   t3   .
Tuesday    Music          5   t4   .
Tuesday    Soc. Studies   6   t5   .
Wednesday  Mathematics    1   w1   .
Wednesday  Language       2   w2   .
Wednesday  Soc. Studies   3   w3   .
Wednesday  Phys. Ed.      5   w4   .
Wednesday  Science        6   w5   .
Thursday   Language       1   th1  .
Thursday   Mathematics    2   th2  .
Thursday   Science        3   th3  .
Thursday   Phys. Ed.      5   th4  .
Thursday   Soc. Studies   6   th5  .
Friday     Mathematics    1   f1   .
Friday     Language       2   f2   .
Friday     Soc. Studies   3   f3   .
Friday     Library        5   f4   .
Friday     Science        6   f5   .
;
```

```
data anno;
   /* Set up required variable lengths, etc. */
   length function color style   $8;
   length xsys ysys hsys         $1;
   length when position          $1;

   xsys     = '2';
   ysys     = '2';
   hsys     = '4';
   when     = 'a';

   function = 'label   ';
   x = 4;
   size = 2;
   position = '5';
   y=5; TEXT='L'; output;
   y=4; TEXT='U'; output;
   y=3; TEXT='N'; output;
   y=2; TEXT='C'; output;
   y=1; TEXT='H'; output;
   run;

pattern1 v=s c=pink;
title 'Class Schedule: 2003-2004';
footnote j=l h=2 '  Teacher: Mr. A. Smith Hall'
         j=r h=2 'Room: 107  ';
proc netdraw graphics data=schedule anno=anno;
   actnet / act=daytime
            succ=msucc
            id=(class)
            nodefid nolabel
            zone=day
            align=time
            useformat
            linear
            pcompress
            coutline=black
            hmargin = 2 vmargin = 2
            htext=2;
   run;
```

Output 9.16.1 Use of the Annotate Facility



Class Schedule: 2003-2004

Day \ Time	9:00 - 10:00	10:00 - 11:00	11:00 - 12:00	12:00 - 1:00	1:00 - 2:00	2:00 - 3:00
Monday	Mathematics	Language	Soc. Studies	L	Art	Science
Tuesday	Language	Mathematics	Science	U	Music	Soc. Studies
Wednesday	Mathematics	Language	Soc. Studies	N	Phys. Ed.	Science
Thursday	Language	Mathematics	Science	C	Phys. Ed.	Soc. Studies
Friday	Mathematics	Language	Soc. Studies	H	Library	Science

Teacher: Mr. A. Smith Hall

Room: 107

Example 9.17: AOA Network Using the Annotate Facility

This example illustrates the use of the Annotate Facility to draw an Activity-on-Arc network. First, PROC NETDRAW is invoked with explicit node positions for the vertices of the network. The ALIGN= and ZONE= options are used to provide horizontal and vertical axes as a frame of reference. The resulting diagram is shown in Output 9.17.1.

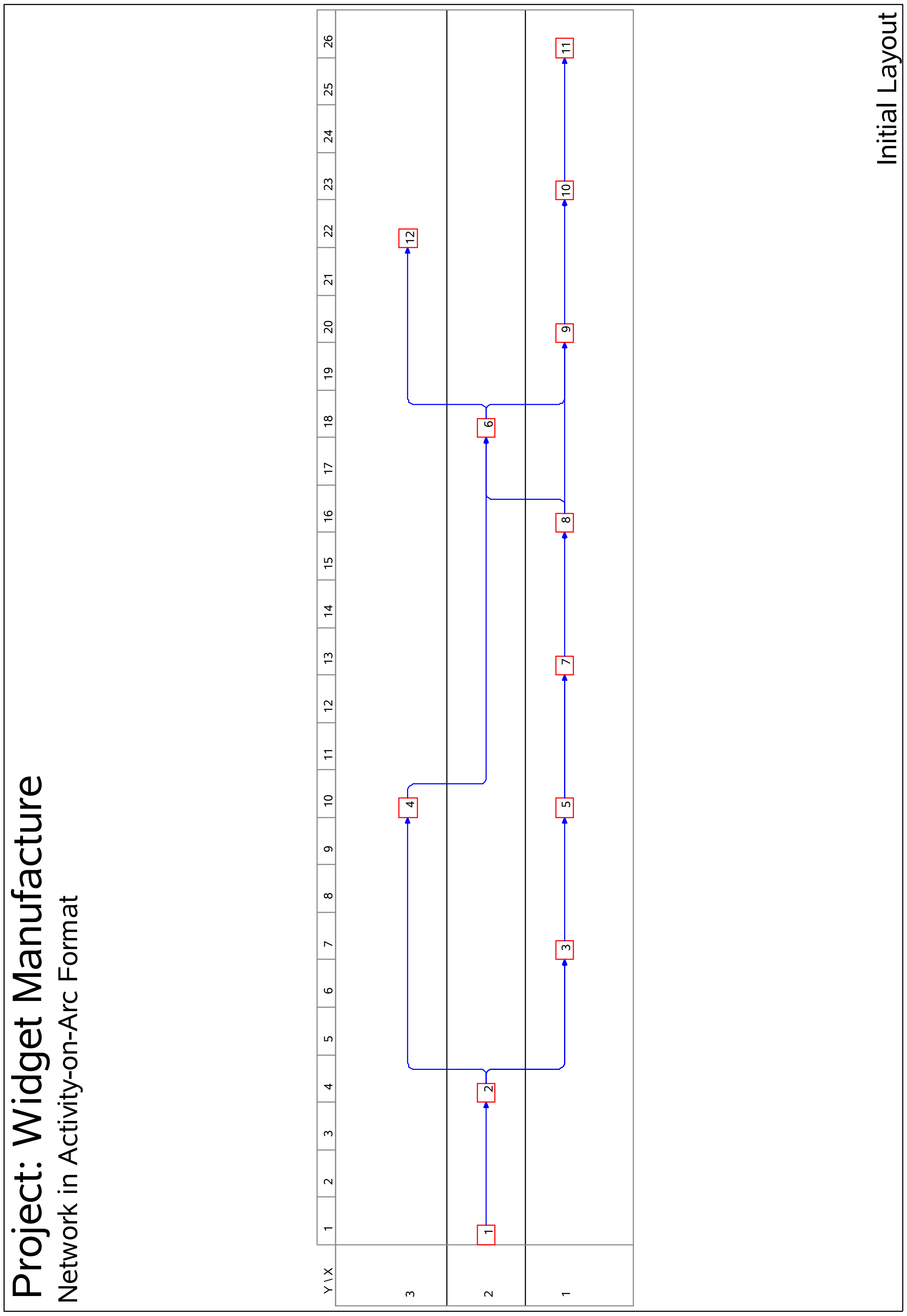
```
data widgaoa;
   format task $12. ;
   input task & days tail head _x_ _y_;
   datalines;
Approve Plan   5   1   2   1  2
Drawings      10   2   3   4  2
Study Market   5   2   4   4  2
Write Specs    5   2   3   4  2
Prototype     15   3   5   7  1
Mkt. Strat.   10   4   6  10  3
Materials     10   5   7  10  1
Facility      10   5   7  10  1
Init. Prod.   10   7   8  13  1
```

```
Evaluate      10   8   9  16  1
Test Market   15   6   9  18  2
Changes        5   9  10  20  1
Production     0  10  11  23  1
Marketing      0   6  12  19  2
Dummy          0   8   6  16  1
.              .  11   .  26  1
.              .  12   .  22  3
;

pattern1 v=e c=red;
title j=l h=3 ' Project: Widget Manufacture';
title2 j=l h=2 ' Network in Activity-on-Arc Format';
footnote j=r h=2 'Initial Layout ';

proc netdraw graphics data=widgaoa out=netout;
   actnet / act=tail
            succ=head
            id=(tail)
            align=_x_
            zone=_y_
            ybetween = 10
            nodefid
            nolabel
            pcompress
            htext=2;
   label _y_=' Y \ X ';
run;
```

In Output 9.17.1, the arc leading from vertex ‘4’ to vertex ‘6’ has two turning points: (10.5, 3) and (10.5, 2). Suppose that you want the arc to be routed differently, to provide a more symmetric diagram. The next DATA step creates a data set, NETIN, which changes the *x* coordinates of the turning points to 16.5 instead of 10.5. Further, two Annotate data sets are created: the first one labels the nodes outside the boxes, either to the top or to the bottom, and the second one sets labels for the arcs. PROC NETDRAW is then invoked with the combined Annotate data set to produce the diagram shown in Output 9.17.2.

```
data netin;
  set netout;
  if _from_=4 and _to_=6 and _seq_>0 then _x_=16.5;
  run;

data anno1;
   set netout;
   if _seq_=0;
   /* Set up required variable lengths, etc. */
   length function color style   $8;
   length xsys ysys hsys         $1;
   length when position          $1;
   length TEXT                   $12;
   xsys     = '2';
   ysys     = '2';
```

Output 9.17.1 Activity-on-Arc Format



```
   hsys     = '4';
   when     = 'a';
   function = 'label   ';
   size = 2;
   position = '5';
   TEXT = left(put(tail, f2.));
   x=_x_;
   if _y_ = 1 then y=_y_-.3;
   else            y=_y_+.5;
   run;

data anno2;
   /* Set up required variable lengths, etc. */
   length function color style   $8;
   length xsys ysys hsys         $1;
   length when position          $1;
   length TEXT                   $12;
   xsys     = '2';
   ysys     = '2';
   hsys     = '4';
   when     = 'a';
   function = 'label   ';
   size = 2;
   position = '5';
   x=2.5;  y=1.8;  TEXT='Approve Plan'; output;
   x=5.5;  y=.8;   TEXT='Drawings';     output;
   x=5.7;  y=1.4;  TEXT='Write Specs';  output;
   x=7;    y=3.4;  TEXT='Study Market';  output;
   x=8.5;  y=.8;   TEXT='Prototype';    output;
   x=11.5; y=1.4;  TEXT='Facility';     output;
   x=11.5; y=.8;   TEXT='Materials';    output;
   x=14.5; y=.9;   TEXT='Init. Prod';   output;
   x=13.5; y=3.4;  TEXT='Mkt. Strat.';  output;
   x=18;   y=.8;   TEXT='Evaluate';     output;
   x=21.5; y=.8;   TEXT='Changes';      output;
   x=24.5; y=.8;   TEXT='Production';   output;
   x=20;   y=3.4;  TEXT='Marketing';    output;
   position=6;
   x=16.6; y=1.5;  TEXT='Dummy';        output;
   x=18.6; y=1.5;  TEXT='Test Market';  output;
   ;

data anno;
   set anno1 anno2;
   run;

footnote j=r h=2 'Annotated and Modified Layout ';
pattern1 v=s c=red;

proc netdraw graphics data=netin anno=anno;
   actnet / nodefid
            nolabel
            boxwidth=1
            pcompress
```

```
               novcenter
               vmargin=20
               xbetween=10;
run;
```

Example 9.18: Branch and Bound Trees

This example illustrates a nonstandard use of PROC NETDRAW. The TREE option in PROC NETDRAW is used to draw a branch and bound tree such as one that you obtain in the solution of an integer programming problem. Refer to Chapter 5, “The LP Procedure” (*SAS/OR User’s Guide: Mathematical Programming Legacy Procedures*), for a detailed discussion of branch and bound trees. The data used in this example were obtained from one particular invocation of PROC LP.

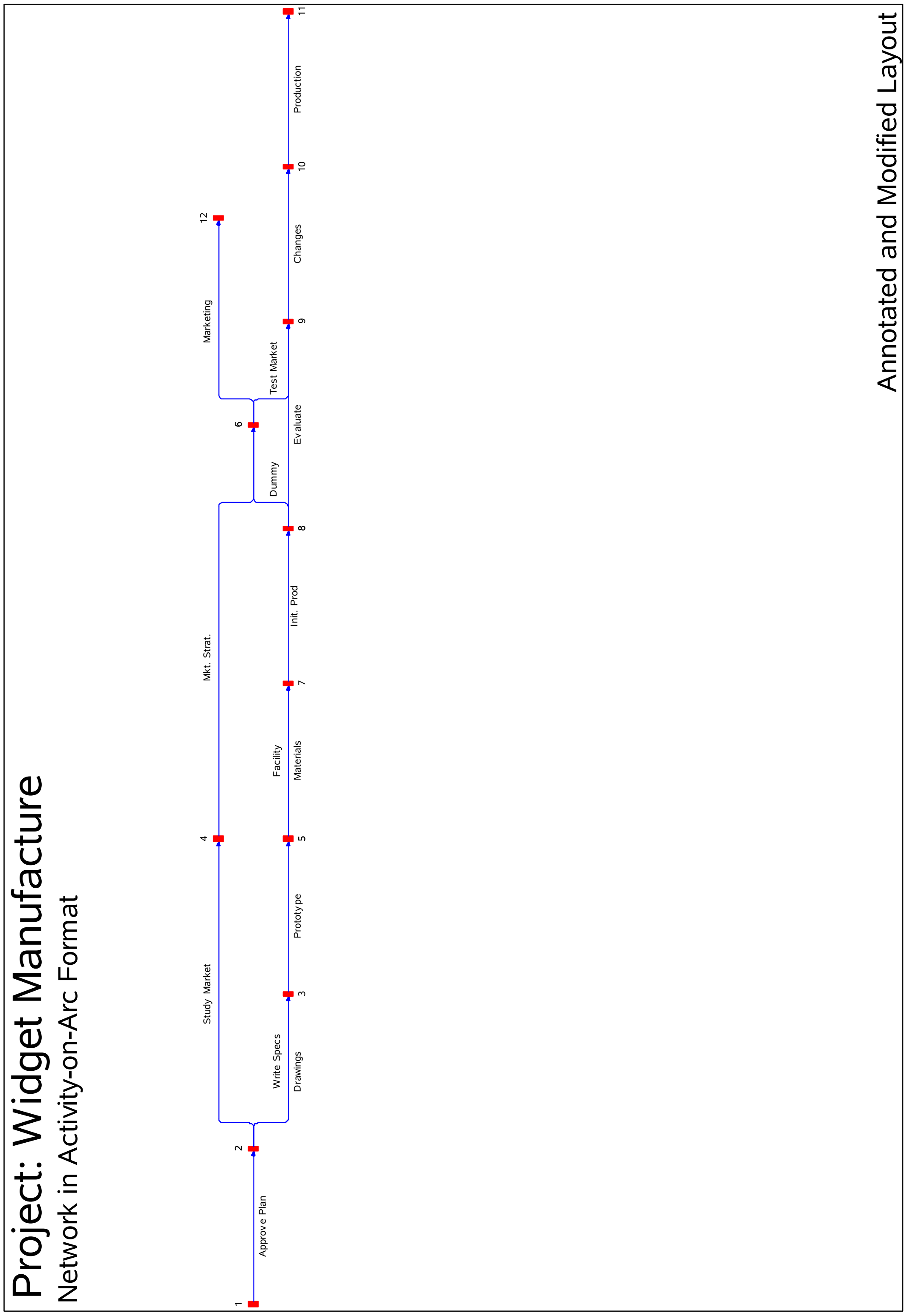
The data set NET (created in the following DATA step) contains information pertaining to the branch and bound tree. Each observation of this data set represents a particular iteration of the integer program, which can be drawn as a node in the tree. The variable node names the problem. The variable object gives the objective value for that problem. The variable problem identifies the *parent* problem corresponding to each node; for example, since the second and the seventh observations have problem equal to ‘-1’ and ‘1’, respectively, it indicates that the second and the seventh problems are derived from the first iteration. Finally, the variable _pattern specifies the pattern of the nodes based on the status of the problem represented by the node.

```
data net;
   input node problem cond $10. object;
   if cond="ACTIVE"          then _pattern=1;
   else if cond="SUBOPTIMAL" then _pattern=2;
   else                           _pattern=3;
datalines;
 1       0     ACTIVE         4
 2      -1     ACTIVE         4
 3       2     ACTIVE         4
 4      -3     ACTIVE 4.3333333
 5       4 SUBOPTIMAL         5
 6       3   FATHOMED 4.3333333
 7       1     ACTIVE         4
 8      -7     ACTIVE         4
 9      -8   FATHOMED 4.3333333
10       8   FATHOMED 4.3333333
11       7     ACTIVE         4
12     -11   FATHOMED 4.3333333
13      11   FATHOMED       4.5
;
```

The next DATA step (which creates the data set LOGIC) uses this child-parent information to format the precedence relationships as expected by PROC NETDRAW. Next, the two data sets are merged together to create the Network input data set (BBTREE) for PROC NETDRAW. The ID variable in the data set BBTREE is formatted to contain both the iteration number and the objective value.

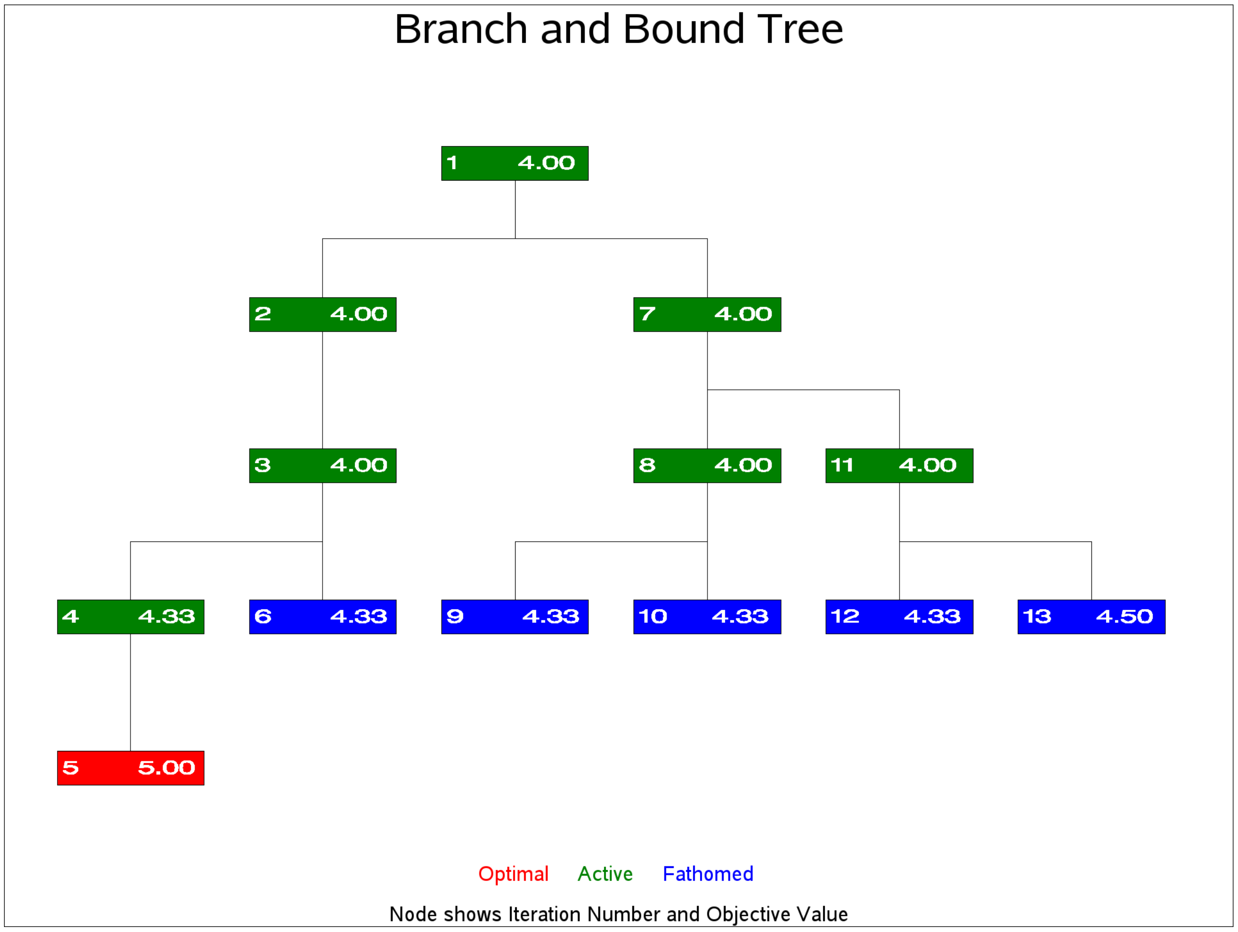
Finally, PROC NETDRAW is invoked with the TREE, ROTATE, and ROTATETEXT options to produce a branch and bound tree shown in Output 9.18.1. Note that the ROTATE and ROTATETEXT options produce a rotated graph with a top-down orientation.

Output 9.17.2 Activity-on-Arc Format: Annotated Diagram



```
/* set precedence relationships
   using child-parent information */
data logic;
   keep node succ;
   set net(firstobs=2);
   succ=node;
   node=abs(problem);
   run;

proc sort data=logic;
   by node;
   run;

/* combine the logic data and the node data */
/* set ID values                             */
data bbtree;
  length id $ 9;
  merge logic net; by node;
  if node < 10 then id=put(node,1.)||put(object,f8.2);
  else              id=put(node,2.)||put(object,f7.2);
  run;

goptions border rotate=portrait;
pattern1 v=s c=green;
pattern2 v=s c=red;
pattern3 v=s c=blue;

title h=3    j=c 'Branch and Bound Tree';
title2    ' ';
footnote1 h=1.5 j=c c=red   'Optimal  '
                    c=green '   Active  '
                    c=blue  '   Fathomed ';
footnote2 ' ';
footnote3 h=1.5 ' Node shows Iteration Number and Objective Value ';
proc netdraw data=bbtree graphics out=bbout;
   actnet /activity=node
           successor=succ
           id=(id)
           nodefid
           nolabel
           ctext=white
           coutline=black
           carcs=black
           xbetween=15
           ybetween=3
           compress
           font=swiss
           rectilinear
           tree
           rotate
           rotatetext
           arrowhead=0
           htext=2;
   run;
```

Output 9.18.1 Branch and Bound Tree

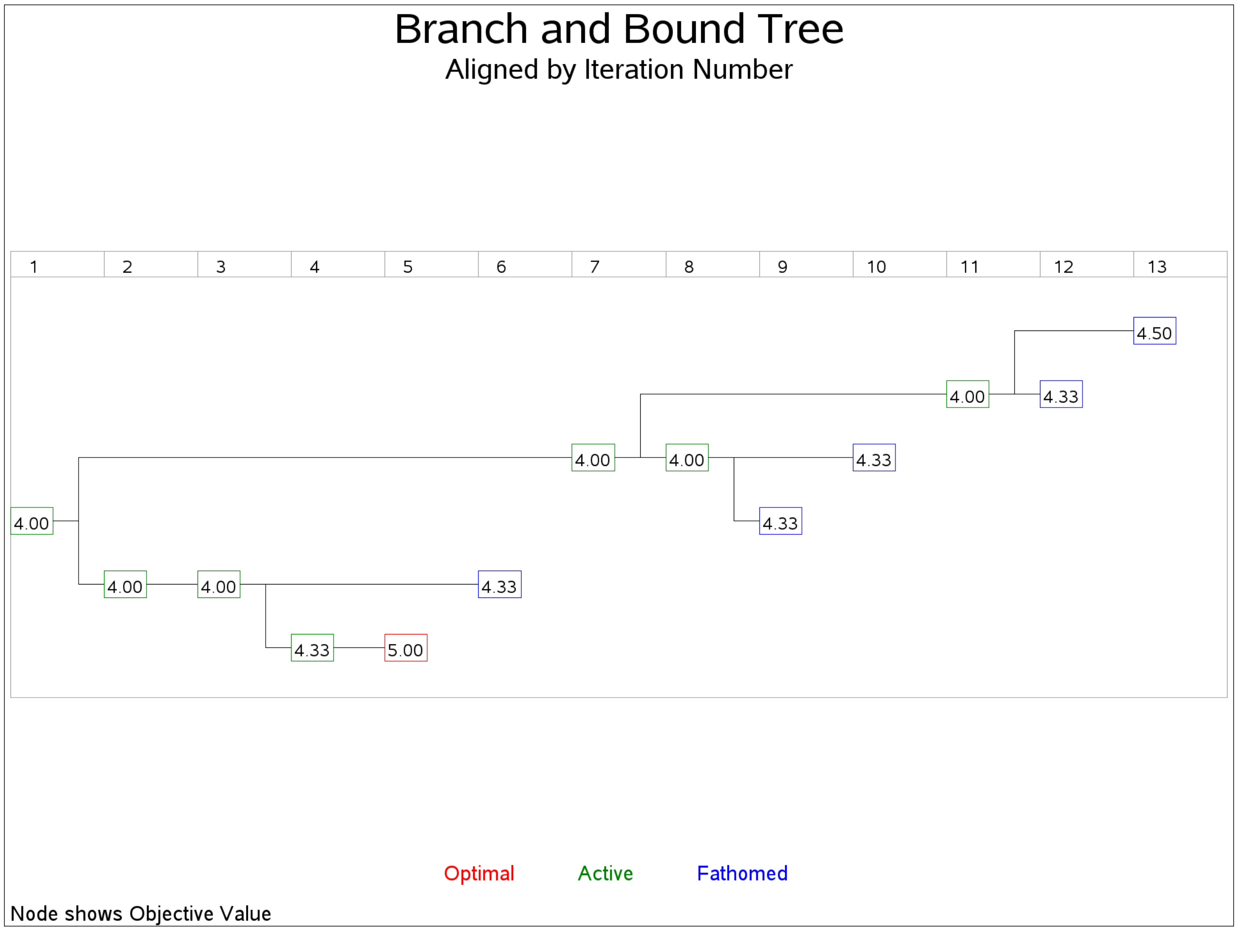


In the next invocation, PROC NETDRAW uses a modified layout of the nodes to produce a diagram where the nodes are aligned according to the iteration number. The following program uses the Layout data set produced in the previous invocation of PROC NETDRAW. The same *y* coordinates are used; but the *x* coordinates are changed to equal the iteration number. Further, the ALIGN= specification produces a time axis that labels each level of the diagram with the iteration number. Each node is labeled with the objective value. The resulting diagram is shown in Output 9.18.2.

```
data netin;
   set bbout;
   if _seq_ = 0; drop _seq_ ;
   _x_ = _from_;
   id = substr(id, 3);
   run;

goptions rotate=landscape;
title h=3   'Branch and Bound Tree';
title2 h=2  'Aligned by Iteration Number';
footnote1 h=1.5 j=c  c=red   'Optimal      '
                     c=green '      Active      '
                     c=blue  '      Fathomed ';
footnote2 ' ';
```

```
footnote3 j=l h=1.5 ' Node shows Objective Value ';
pattern1 v=e c=green;
pattern2 v=e c=red;
pattern3 v=e c=blue;
proc netdraw data=netin graphics;
   actnet /id=(id)
           ctext=black
           carcs=black
           align = _from_
           frame
           pcompress
           rectilinear
           arrowhead=0
           nodefid
           nolabel
           htext=2.5
           xbetween=8;
   run;
```

Output 9.18.2 Branch and Bound Tree: Aligned by Iteration Number



Statement and Option Cross-Reference Tables

The next two tables reference the options in the NETDRAW procedure that are illustrated by the examples in this section. Note that all the options are specified on the ACTNET statement.

Table 9.9 Options Specified in Examples 5.1–5.9

Option	1	2	3	4	5	6	7	8	9
ACTIVITY=	X	X	X	X	X	X	X	X	X
ALIGN=									X
ARROWHEAD=						X			
AUTOREF									X
BOXHT=						X	X		
BOXWIDTH=					X				X
CARCS=		X	X	X				X	X
CCRITARCS=			X	X				X	X
CCRITOUT=			X					X	
CENTERID						X			
COMPRESS		X		X					
COUTLINE=			X					X	
CREF=									X
CTEXT=			X						
DATA=	X	X	X	X	X	X	X	X	X
DP							X		
DURATION=	X		X	X			X		
FONT=				X					
FRAME									X
GRAPHICS		X	X	X	X	X	X	X	X
HMARGIN=									X
HPAGES=					X				X
HTEXT=		X			X	X	X	X	X
HTRACKS=							X		
ID=					X			X	X
LAG=						X			
LINEPRINTER	X								
LREF=									X
LWCRIT=			X						
LWIDTH=		X	X						
NODEFID					X			X	X
NOLABEL					X			X	X
NOPAGENUMBER					X				X
NOVCENTER				X					X
PATTERN=								X	
PCOMPRESS				X		X	X	X	X
SEPARATEARCS			X	X	X	X	X	X	X
SHOWBREAK									X
SHOWSTATUS								X	X

Table 9.9 (continued)

Option	1	2	3	4	5	6	7	8	9
SUCCESSOR=	X	X	X	X	X	X	X	X	X
TIMESCALE									X
VMARGIN=									X
VPAGES=					X				X
XBETWEEN=						X	X		
YBETWEEN=					X	X		X	X

Table 9.10 Options Specified in Examples 5.10–5.18

Option	10	11	12	13	14	15	16	17	18
ACTIVITY=	X	X	X	X	X	X	X	X	X
ALIGN=							X	X	X
ANNOTATE=							X	X	
ARROWHEAD=						X			X
AUTOREF		X							
BOXHT=				X					
BOXWIDTH=	X	X						X	
BREAKCYCLE			X						
CARCS=	X	X	X	X	X	X			X
CAXIS=		X							
CCRITARCS=		X							
CENTERID			X	X	X	X			
CENTERSUBTREE						X			
COMPRESS	X								X
COUTLINE=			X	X			X		X
CREF=		X							
CREFBRK=	X								
CTEXT=			X	X	X	X			X
DATA=	X	X	X	X	X	X	X	X	X
DP	X								
DURATION=	X								
FONT=	X								X
FRAME	X								X
GRAPHICS	X	X	X	X	X	X	X	X	X
HMARGIN=			X				X		
HTEXT=	X	X	X	X	X	X	X	X	X
ID=	X	X	X	X		X	X	X	X
LINEAR	X	X					X		
LREF=		X							
LREFBRK=	X								
MININTERVAL=		X							
NOARROWFILL			X						
NODEFID	X	X	X	X		X	X	X	X
NOLABEL	X	X	X	X		X	X	X	X
NOVCENTER	X		X			X		X	

Table 9.10 (continued)

Option	10	11	12	13	14	15	16	17	18
OUT=				X	X			X	X
PATTERN=			X						
PCOMPRESS	X	X	X	X	X	X	X	X	X
RECTILINEAR			X			X			X
REFBREAK	X								
ROTATE									X
ROTATETEXT									X
SEPARATEARCS	X	X		X	X				
SEPARATESONS						X			
SUCCESSOR=	X	X	X	X	X	X	X	X	X
TIMESCALE	X	X							
TREE						X			X
USEFORMAT							X		
VMARGIN=	X		X				X	X	
XBETWEEN=			X			X		X	X
YBETWEEN=			X	X	X	X		X	X
ZONE=		X					X	X	
ZONEPAT		X							
ZONESPACE		X							

References

Even, S. (1979), *Graph Algorithms*, Rockville, MD: Computer Science Press.

Kuhfeld, W. F. (2010), *Statistical Graphics in SAS: An Introduction to the Graph Template Language and the Statistical Graphics Procedures*, Cary, NC: SAS Institute Inc.

Subject Index

Syntax Index