
Subject: Re: Is and PV-WAVE problem on HP-UX 10.20

Posted by [stroyan](#) on Wed, 27 Jan 1999 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Franz Dollinger (Franz.Dollinger@mchp.siemens.de) wrote:

|after a patch bundle (10.20.40) had been installed to our
|HP 9000/735's on HP-UX 10.20, I run into problems
|with long command lines.
|The problems didn't occur before the installation.

...

|Any idea on how to overcome that limitation?

Recent patch bundles on 10.20 include the ability to have very long argument lists. The feature is controlled by a configurable kernel parameter, "large_ncargs_enabled". If you have the right patches installed you can set that parameter to 1 in sam under the "Kernel Configuration" -> "Kernel Parameters" section.

--

Mike Stroyan, mike_stroyan@fc.hp.com

Subject: Re: Is and PV-WAVE problem on HP-UX 10.20

Posted by [thompson](#) on Wed, 27 Jan 1999 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Franz.Dollinger@mchp.siemens.de (Franz Dollinger) writes:

> Hi,

> after a patch bundle (10.20.40) had been installed to our
> HP 9000/735's on HP-UX 10.20, I run into problems
> with long command lines.
> The problems didn't occur before the installation.

> calling any program with a long list of files
> e.g.:
> ls vvv_*
> where vvv_* stands for more than 1000 files results
> in
> arg list too long

> Same behaviour for the findfile routine of PV-WAVE.
> It got the same limitation since the installation
> of the patch bundle.

> Any idea on how to overcome that limitation?

> thanks

> bye

> Franz

This is a limitation imposed by the Unix shell. The following routine uses find instead of ls to overcome this.

William Thompson

```
=====
=====
;+
; Project   : SOHO - CDS
;
; Name      : FIND_FILE()
;
; Purpose   : Fixing builtin FINDFILE() problem
;
; Explanation : The builtin FINDFILE() function has problems on some unixes
;               whenever *a lot* of files are matching the file
;               specification. This is due to the fact that filename expansion
;               is done by the shell *before* interpreting a command. Too many
;               files cause too long commands, which are not accepted. This
;               causes FINDFILE() to return an empty list of candidates.
;
;               FIND_FILE tries the builtin function first, and whenever the
;               returned list of files is empty, it tries to recheck through
;               spawning a "find" command.
;
;               Since FINDFILE doesn't discriminate between directories, links
;               and files, this function will not do this either.
;
;               Under unix, however, calls like FINDFILE("*") returns the
;               unfiltered output of the shell command "ls *", including
;               colon-terminated lines for each subdirectory matching the
;               specification and empty lines separating each subdirectory
;               listing. Such silly effects are not implemented in the "find"
;               version. Be warned, however, that these effects are present
;               when the builtin function does not "fail" due to a too long
;               file list.
;
;               It is possible (under unix) to use the "find" method as
;               default by setting the keyword /USEFIND (no effect under other
;               operating systems).
;
```

```

; Use      : files = find_file(file_specification)
;
; Inputs   : file_specification : A scalar string used to find
;           : files. See FINDFILE()
;
; Opt. Inputs : None.
;
; Outputs   : Returns a list of files or a blank string if none found.
;
; Opt. Outputs:
;
; Keywords  : COUNT : Returns the number of files
;
;           : USEFIND : Always use a spawned "find" command under unix.
;           :           No effect under other operating systems.
;
; NODOT : Apply a filter to the results from find to prevent
; finding the directory itself in a large file expansion.
; eg 'find_file,"foo/*"' returns ("foo/", "foo/a",...)
; but 'find_file,"foo/*",/nodot' returns
; ("foo/a","foo/b",...) without the leading "foo/".
; This behavior is closer to the behavior of findfile()
; without the long-directory braindamage. It is
; *not* the default so as not to break heritage
; code that uses find_file().
;
; Calls    : BREAD_FILE, FINDFILE, SPAWN
;
; Common   : None
;
; Restrictions: As for FINDFILE
;
; Side effects: None, hopefully
;
; Category  : Utilities, Operating_system
;
; Prev. Hist. : Lots of problems with FINDFILE is hopefully history.
;
; Written    : S.V.H. Haugan, UiO, 12 April 1996
;
; Modified   : Version 2, SVHH, 10 June 1996
;             Moved the CD,curr_path command to avoid
;             returns without resetting path.
;             Version 3, SVHH, 26 June 1996
;             Took away the -type f argument to find, added
;             /USEFIND keyword.
;             : Added /nodot keyword C. DeForest 9-August-1998
;
;

```

; Version : 3, 26 June 1996

;-

FUNCTION find_file,file_specification,count=count,usefind=usefind,nod ot=nodot

count = 0

use_find = KEYWORD_SET(usefind) AND os_family() EQ 'unix'

IF NOT use_find AND N_PARAMS() EQ 0 THEN BEGIN

result = findfile(count = count)

RETURN,result ; Unix doesn't have problems with this

END

IF N_PARAMS() EQ 0 THEN file_specification = '*'

IF file_specification EQ " THEN file_specification = '*'

IF NOT use_find THEN result = findfile(file_specification,count=count) \$

ELSE count = 0

;; Check for problems

IF count EQ 0 AND os_family() EQ 'unix' THEN BEGIN

file = file_specification

break_file,file,disk,dir,filnam,ext

;; Check if directory exists

IF dir NE " THEN BEGIN

IF (findfile(dir))(0) eq " THEN RETURN,"

END

;; Temporary switch to that directory

IF dir NE " THEN cd,dir,current=curr_path

IF filnam+ext EQ " THEN filnam = '*'

;; Find all matching

spn = ["find",".", "-name",filnam+ext,"-print"]

spawn,spn,result,/noshell

;; Switch back to original directory

IF dir NE " THEN cd,curr_path

IF result(0) EQ " THEN RETURN," ; None matching, return

;; Get rid of current-directory match, if necessary

if keyword_set(nodot) and result(0) eq '.' then \$

result = result(1:n_elements(result)-1)

```

;; Chop off './'
result = STRMID(result,2,1000)

;; Chip out subdirectories (for some reason, the -prune option doesn't
;; work properly, so I have dropped using it).

ix = WHERE(STRPOS(result,'/') EQ -1,count)
IF count EQ 0 THEN RETURN,"

;; Put back the specified (not full) path
result = dir + result(ix)
END

RETURN,result

END

```

```

.....
; End of 'findfile.pro'.
.....

```

Subject: Re: Is and PV-WAVE problem on HP-UX 10.20
 Posted by [R.Bauer](#) on Wed, 27 Jan 1999 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Franz Dollinger wrote:

```

> Hi,
>
> after a patch bundle (10.20.40) had been installed to our
> HP 9000/735's on HP-UX 10.20, I run into problems
> with long command lines.
> The problems didn't occur before the installation.
>
> calling any program with a long list of files
> e.g.:
> ls vv_*
> where vv_* stands for more than 1000 files results
> in
>   arg list too long
>
> Same behaviour for the findfile routine of PV-WAVE.
> It got the same limitation since the installation
> of the patch bundle.
>
> Any idea on how to overcome that limitation?
>

```

> thanks
>
> bye
> Franz
>
> --
> Standard->disclaimer
> (You didn't really think the opinions expressed herein
> are in any way related to my employer?)

The only way i know is to use findfile is result=findfile().
This works!
You are able to use strpos to do the selection you want.

R.Bauer
