## NAME

gvpack – merge and pack disjoint graphs

## SYNOPSIS

**gvpack** [ **–nguv?** ] [ **–m***margin* ] [ **–array***opts* ] [ **–o***outfile* ] [ **–s***graph_name* ] [ **–G***name***=***value* ] [ *files* ]

## DESCRIPTION

**gvpack** reads in a stream of graphs, combines the graphs into a single layout, and produces a single graph serving as the union of the input graphs. The input graphs must be in dot format, and must have all necessary layout information. Acceptable input is produced by applying a Graphviz layout program, such as **dot** or **neato**, with no **–T** flag.

By default, the packing is done at the cluster level. Thus, parts of one graph will not intrude into any top-level clusters or overlap any nodes or edges of another.

The output of **gvpack** can be used to produce concrete output by applying **neato –s –n2** with the desired **–T** flag.

## OPTIONS

The following options are supported:

**–g** Combines the graphs at the graph level. This uses more space, but prevents parts of one graph from occurring between parts of another.

**–array***[_flags][n]*
Combines the graphs at the graph level, placing them in an array. By default, the layout is done in row-major order. The number of columns used is roughly the square root of the number of graphs. If the optional integer *n* is supplied, this indicates the number of columns to use.

If optional flags are supplied, these consist of an underscore followed
by any of the letters "c", "t", "b", "l", "r", "u" or "i". If "c" is supplied, the graphs are packed in column-major order, in which case a final integer specifies the number of rows. The flags "t", "b", "l", "r" indicate that components are aligned along the top, bottom, left or right, respectively. By default, the insertion order is determined by sorting the graphs by size, largest to smallest. If the "u" flag is set, the graphs are sorted based on the non-negative integer *sortv* attribute attached to each graph. The "i" flag indicates that no sorting is done, with the graphs inserted in input order.

**–G***name***=***value*
Specifies attributes to be added to the resulting union graph. For example, this can be used to specify a graph label.

**–m***margin*
Packs the graphs allowing a margin of *output* points around the parts.

**–n** Combines the graphs at the node level. Clusters are ignored in the packing.

**–o***output*
Prints output to the file *output*. If not given, **gvpack** uses stdout.

**–s***graph_name*
Use *graph_name* as the name of the root graph. By default, "root" is used.

**–u** Don't pack the graphs. Just combine them into a single graph.

**–v** Verbose mode.

**–?** Prints usage information and exit.

## OPERANDS

The following operand is supported:

*files* Names of files containing 1 or more graphs in dot format. If no *files* operand is specified, the standard input will be used.

## RETURN CODES

**gvpack** returns **0** if there were no problems, and non-zero otherwise.

## EXAMPLES

```
ccomps -x abc.gv | dot | gvpack | neato -s -n2 -Tps
```

This pipeline decomposes the graph in *abc.gv* into its connected components, lays out each using **dot**, packs them all together again, and produces the final drawing in PostScript. Of course, there is nothing to prevent one from using different layouts for each component.

## BUGS

All the input graphs must be directed or undirected.

An input graph should not have a label, since this will be used in its layout. Since **gvpack** ignores root graph labels, resulting layout may contain some extra space.

**gvpack** unsets the bounding box attribute of all non-cluster subgraphs.

## AUTHORS

Emden R. Gansner <erg@research.att.com>

## SEE ALSO

gvpr(1), dot(1), neato(1), twopi(1), ccomps(1), libpack(3)