NAME

bcg, BCG – Binary Coded Graphs: binary file format for labelled transition systems

DESCRIPTION

The acronym *BCG* stands for *B*inary *C*oded *G*raphs. It refers both to a format and a software environment.

The *BCG format* is a computer representation for labelled transition systems, Kripke structures, and, more generally, state/transition models (hereafter called *graphs*) which are generated from higher-level models of concurrent systems.

Compared to other existing formats with similar purposes, the BCG format combines several advantages:

- The BCG format is independent from any particular model-based verification technique; it can be used either by tools performing graph comparison and reduction modulo equivalence relations, or by tools checking properties expressed in temporal logics. It can also be used in other contexts: graph exploration, graph drawing, etc.
- The BCG format is independent from the language used to describe the concurrent system to be analyzed. It was designed so as to allow the representation of graphs generated from various concurrent languages.
- The BCG format provides sophisticated mechanisms to keep track of source-level information. This is especially useful, since it is possible to establish a connection between the behaviour of a BCG graph and the source program from which the BCG graph was generated.
- The BCG format uses a binary representation of the LTS together with ad-hoc compression algorihms, leading to very compact files. Graphs with millions of states and transitions can be represented using only a few megabytes of disk space.

Moreover, the BCG format is supported by an efficient and extensive software implementation, the *BCG environment*.

APPLICATION TOOLS

In the BCG environment, the programs dealing with graphs encoded in the BCG format are called ‘‘application tools’’. These programs can be either binary programs or shell-scripts.

The following application tools are currently available (see the corresponding manual pages):

bcg_cmp(LOCAL)
Compare normal, probabilistic, or stochastic BCG graphs

bcg_draw(LOCAL)
Display BCG graphs

bcg_edit(LOCAL)
Edit PostScript files generated by **bcg_draw**(LOCAL)

bcg_info(LOCAL)
Print information about BCG graphs

bcg_io(LOCAL)
Convert graphs from and to the BCG format

bcg_graph(LOCAL)
Generate various kinds of useful BCG graphs

bcg_labels(LOCAL)
Hide/rename labels of graph encoded to the BCG format

bcg_lib(LOCAL)
Generate dynamic libraries for BCG graphs

bcg_min(LOCAL)

Minimize of normal, probabilistic, or stochastic BCG graphs

bcg_open(LOCAL)
Execute OPEN/CAESAR application programs on BCG graphs

bcg_steady(LOCAL)
Perform steady-state analysis on Markov chains encoded in the BCG format

bcg_transient(LOCAL)
Perform transient analysis on Markov chains encoded in the BCG format

Application Programming Interfaces for reading and/or writing BCG files from C or C++ programs are described in the **bcg_read**(LOCAL) and **bcg_write**(LOCAL) manual pages.

BCG FILES

Graphs encoded in the BCG format are stored in files having the **.bcg** suffix. These files contain binary data, which are not directly readable by the user.

There are currently several ways of generating a BCG file:

- The **caesar**(LOCAL) tool can generate a BCG graph corresponding to a LOTOS description.
- The **bcg_io**(LOCAL) application program can be used to translate an existing graph into a BCG graph.
- The **generator**(LOCAL) and **reductor**(LOCAL) programs of the OPEN/CAESAR environment can be used to generate a BCG graph.

HOW TO CREATE A BCG GRAPH

If you want to create a BCG graph by yourself, please consult the **bcg_write**(LOCAL) manual page.

HOW TO READ A BCG GRAPH

If you want to read a BCG graph by yourself, please consult the **bcg_read**(LOCAL) manual page.

STATIC LIBRARIES

Application tools rely on libraries of compiled object code provided with the BCG environment. These libraries are called ‘‘static libraries’’. They are not directly visible by the user.

Caution: the contents of static libraries are undocumented and subject to future changes.

DYNAMIC LIBRARIES

In order to access the informations contained in BCG files, application programs have to generate various files containing compiled object code.

These files are called ‘‘dynamic libraries’’ since they depend on BCG files to be accessed, which is not the case of ‘‘static libraries’’.

For a given BCG file *filename***.bcg**, the corresponding dynamic libraries are files named *filename***@1.o**, *filename***@2.o**, etc. They are stored in the same directory as the BCG file itself.

Dynamic libraries can be safely removed: if they are necessary, they will be generated again by application tools. However, it is advised not to delete them systematically, since their generation takes a certain amount of time.

Caution: the contents of dynamic libraries are undocumented and subject to future changes.

ENVIRONMENT VARIABLES

Environment variables are defined and used according to UNIX conventions: see the **set** and **unset** commands of the UNIX shells **sh**(1) and **csh**(1), and also the manual pages for **environ**(5), **getenv**(3), and

putenv(3).

Environment variables are taken into account by all application tools.

The following environment variables are defined:

$CADP_CC
If this variable is set, its value determines the name of the C compiler that will be invoked by application tools. See file **$CADP/INSTALLATION_2** for detailed information about this variable. If this variable is unset, the script-shell **$CADP/src/com/cadp_cc** will automatically determine the C compiler to be used by default.

$CADP_TMP
If this variable is set, its value determines the directory in which temporary files will be created by application tools. If this variable is unset, it is given the default value '**/tmp**'. The names of temporary files are always be prefixed with the string "**bcg_**".

$BCG
If this variable is set, its value should reference the directory in which the BCG package is installed. By default, this variable is supposed to be unset: the BCG package is normally installed in the directory referenced by the environment variable **$CADP**. Setting this variable should be avoided in official distributions of the BCG package, since it may cause problems.

$BCG_DEBUG
If this variable is set, diagnostic messages (which can be helpful in debugging) are displayed on the standard error stream. By default, this variable is supposed to be unset. Setting this variable has no effect in official distributions of the BCG package.

GENERAL OPTIONS

There exist command-line options which are common to many, if not all, application tools. Their meaning is the same in all the applications tools that support these options.

Unless stated otherwise, general options can appear at any place on the command-line when an application tool is invoked.

The following general options are currently available:

−version
Display the current version number of the application tool and stop. To be effective, this option should occur as the first argument on the command line. Subsequent options, if any, will be discarded.

−create Force the dynamic library to be created, even if it already exists in the current directory and if it is up-to-date. Not a default option.

−update
Do not create the dynamic library if it already exists in the current directory and if it is up-to-date. Default option.

−remove
Remove the dynamic library after usage. Not a default option.

−cc *options*
Pass *options* to the C compiler when creating dynamic libraries. *options* is a list of compiler options (enclosed in quotes or double quotes). These options are appended to the compiler options, if any, contained in the **$CADP_CC** environment variable (see ENVIRONMENT VARIABLES below). Not a default option.

−tmp *directory*
Use *directory* to store temporary files. If present, this option overrides the environment variable **$CADP_TMP**.

−uncompress
Represent the edges of the graph using an edge area of code 1. This option is only meaningful when a BCG graph is to be created. Not a default option.

−compress
Represent the edges of the graph using an edge area of code 2. This option is only meaningful when a BCG graph is to be created. Default option.

−register *register_size*
Use *register_size* bits to encode registers offsets. This option is only meaningful when a BCG graph is to be created with an edge area of code 2. By default, *register_size* is equal to 4.

−short *short_size*
Use *short_size* bits to encode short branch offsets. This option is only meaningful when a BCG graph is to be created with an edge area of code 2. By default, *short_size* is equal to 1.

−medium *medium_size*
Use *medium_size* bits to encode medium offsets. This option is only meaningful when a BCG graph is to be created with an edge area of code 2. By default, *medium_size* is equal to 3.

−size *register_size short_size medium_size*
This option is equivalent to the simultaneous occurrence of the three following options: **-register** *register_size*, **-short** *short_size*, and **-medium** *medium_size*.

PARTICULAR OPTIONS

Besides general options, application tools can also accept their own options, which are not shared with other application tools. These particular options will be described in the manual pages of each application tools.

DIAGNOSTICS

Application tools share common conventions with respect to diagnostics. Exit status is 0 if everything is alright, 1 otherwise.

AUTHORS

Hubert Garavel (definition of the BCG format) and Renaud Ruffiot (implementation of the BCG environment).

FILES

∗**.bcg**	BCG files
∗**@1.o,** ∗**@2.o, etc.**	dynamic libraries
$CADP/incl/bcg_∗	include files
$CADP/com/bcg_∗	shell-scripts
$CADP/src/com/cadp_cc	C compiler shell
$CADP/src/∗/∗	various source files
$CADP/bin.‘arch‘/bcg_∗	binary programs
$CADP/bin.‘arch‘/libbcg_∗**.a**	static libraries
$CADP/bin.‘arch‘/libBCG_∗**.a**	static libraries
$CADP_TMP/bcg_∗	temporary files

SEE ALSO

bcg_draw(LOCAL), **bcg_edit**(LOCAL), **bcg_graph**(LOCAL), **bcg_info**(LOCAL), **bcg_io**(LOCAL), **bcg_lib**(LOCAL), **bcg_min**(LOCAL), **bcg_open**(LOCAL), **bcg_read**(LOCAL), **bcg_steady**(LOCAL), **bcg_transient**(LOCAL), **bcg_write**(LOCAL).

Additional information is available from the CADP Web page located at http://cadp.inria.fr

Directives for installation are given in files **$CADP/INSTALLATION_***.

Recent changes and improvements to this software are reported and commented in file **$CADP/HISTORY**.

BUGS

Please report bugs to Hubert.Garavel@inria.fr