qsopt_ex-interface

An Interface to QSopt exact LP solver

1.0

03/28/2016

Jayant Apte

Jayant Apte
Email: jayant91089@gmail.com
Homepage: https://sites.google.com/site/jayantapteshomepage/
Address: Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA 19104

Contents

Chapter 1

Introduction

`qsopt_ex-interface` is a GAP package that provides an interface to *QSopt* exact rational linear program solver [ACDE09] by Applegate,Cook,Dash and Espinoza. This is a minimalist package exposing parts of qsopt to GAP. The particular version of QSopt-exact solver this package currently follows is 2.5.10-patch 3 of a fork of the original software maintained by Jon Lund Steffenson [Ste15], which removes certain dependencies and makes the software easier to build. `qsopt_ex-interface` provides a C wrapper qsinterface.c to the solver. It is currently available for Unix/Linux systems running GAP 4.5+.

Chapter 2

Installation

Assuming you already have GAP 4.5+ installed, you can follow the steps below to install the package:

- To get the newest version of `qsopt_ex-interface`, download the .zip archive from https://github.com/jayant91089/qsopt_ex-interface and unpack it using `unzip qsopt_ex-interface-x.zip` in the terminal. Do this preferably inside the *pkg* subdirectory of your GAP 4 installation. It creates a subdirectory called `qsopt_ex-interface`. If you do not know the whereabouts of the *pkg* subdirectory, invoke the following in GAP:

Code

```
GAPInfo.("RootPaths");
```

Look for pkg directory inside any of the paths returned.

- Once unpacked, go to `qsopt_ex-interface` directory and run the install script `unix-install.sh` from the terminal as `sh unix-install.sh`. This locally installs qsopt exact and its dependencies (GMP [GtGdt15],libz and libbz2) in lib and include folders. Alternatively, if you have qsopt-exact and GMP already installed on your system, you can edit the Makefile inside `qsopt_ex-interface` directory so that gcc finds the .so libraries. In latter case, you must manually '\texttt{make all}' from the terminal inside `qsopt_ex-interface` directory.

- Above step creates an executable \texttt{qsi} inside the `qsopt_ex-interface` directory, which serves as the interface. Note that before using the package in GAP, one must edit either the environment variable `LD_LIBRARY_PATH` or the so that \texttt{qsi} finds the locally installed libraries.

- One can now start using `qsopt_ex-interface` by invoking

Code

```
LoadPackage( "qsopt_ex-interface");
```

from within GAP.

Chapter 3

Usage

3.1 Available functions

In this section we shall look at the functions provided by qsopt_ex-interface. `qsopt_ex-interface` allows GAP to communicate with external LP solver process via a stream object of category IsInputOutputStream(). This steam serves as a handle via which one can load/solve/modify linear programs. Note that it is possible to maintain several such steams (and hence LPs) at any given time. However, the gap commands to solve/modify these LPs that currently available in this package are blocking functions.

3.1.1 LoadQSLP

▷ `LoadQSLP(obj, A, b, linrows, qs_exec, optargs)` (function)

Returns: A list

This function loads an LP by invoking external qsopt-exact LP solver process. It accepts following arguments:

- *obj* - Objective function coefficients, provided as a list
- *A* - A list of lists corresponding to constraints
- *b* - Right hand side of constraints
- *linrows* - A list of indices of members of *A* that are equalities
- *qs_exec* - A string describing complete path to 'qsi' executable (including 'qsi')

Returns a list $[s, rval]$ where 's' is a gap object of category IsInputOutputStream() and 'rval' $= 1/-1$ indicates success/failure. If 'rval=1', 's' is ready to be be used to solve linear programs.

3.1.2 LoadQSLPobj

▷ `LoadQSLPobj(s, obj)` (function)

Returns: An integer

This function loads a new objective. It accepts following arguments:

- *s* - gap object of category IsInputOutputStream(), handle to an already loaded LP

- *obj* - Objective function coefficients, provided as a list

Returns an integer 'rval' $= 1/-1$ that indicate success/failure. If 'rval=1', the LP associated with 's' is successfully modified.

3.1.3 SolveQSLP

▷ `SolveQSLP(s)` (function)

Returns: An integer

This function loads an LP by invoking external qsopt-exact LP solver process. It accepts following arguments:

- *s* - gap object of category IsInputOutputStream(), handle to an already loaded LP
- *optargs* - A list of optional arguments. Currently supports only one optional argument, which is an integer specifying simplex variant to use: $optargs = [1]$ for primal simplex, $optargs = [2]$ for dual simplex and $optargs = [3]$ for either

Returns an integer *status* that is the integer returned by `mpq_QSget_status()` function.

3.1.4 FlushQSLP

▷ `FlushQSLP(s)` (function)

Returns:

This function terminates the external processes associated with given LP handle. It accepts following arguments:

- *s* - gap object of category IsInputOutputStream(), handle to an already loaded LP

Returns Nothing

3.1.5 GetQSLPsol_primal

▷ `GetQSLPsol_primal(s)` (function)

Returns: A list

This function obtains the primal solution along with the associated vertex vertex, for the most recently solved LP. It accepts following arguments:

- *s* - gap object of category IsInputOutputStream(), handle to an already loaded LP

Returns A list $[status, val\_rval, val, x\_rval, x]$ if optimal solution exists and a list $[status]$ otherwise. If $status = 1$, *val_rval* and *x_rval* indicate validity of *val* and *x* (valid if 1 and invalid if -1) which are optimal solution and (primal) vertex achieving optimal solution respectively. Other status values correspond to the integer returned by `mpq_QSget_status()` function.

3.1.6 GetQSLPsol_dual

▷ `GetQSLPsol_dual(s)` (function)

Returns: A list

This function obtains the primal solution along with the associated vertex vertex, for the most recently solved LP. It accepts following arguments:

- *s* - gap object of category IsInputOutputStream(), handle to an already loaded LP

Returns A list $[status, val\_rval, val, y\_rval, y]$ if optimal solution exists and a list $[status]$ otherwise. If $status = 1$, *val_rval* and *x_rval* indicate validity of *val* and *x* (valid if 1 and invalid if -1) which are optimal solution and (dual) vertex achieving optimal solution respectively. Other status values correspond to the integer returned by `mpq_QSget_status()` function.

References

[ACDE09] David Applegate, William Cook, Sanjeeb Dash, and Daniel Espinoza. QSopt-ex 2.6 — A computer algebra system for polynomial computations, 2009. 3

[GtGdt15] Torbörn Granlund and the GMP development team. GNU MP: The GNU Multiple Precision Arithmetic Library 6.0.0, 2015. 4

[Ste15] Jon Lund Steffensen. QSopt-ex 2.5.10 patch 3 - a fork adding improvements to the build system, library and a python interface, 2015. 3

Index