C interfaces to GALAHAD HASH

Jari Fowkes and Nick Gould
STFC Rutherford Appleton Laboratory
Wed May 3 2023

Chapter 1

GALAHAD C package hash

1.1 Introduction

1.1.1 Purpose

Set up, insert into, remove from and search a chained scatter table (Williams, CACM 2, 21-24, 1959).

Currently, only the control and inform parameters are exposed; these are provided and used by other GALAHAD packages with C interfaces.

1.1.2 Authors

N. I. M. Gould, STFC-Rutherford Appleton Laboratory, England.

C interface, additionally J. Fowkes, STFC-Rutherford Appleton Laboratory.

Julia interface, additionally A. Montoison and D. Orban, Polytechnique Montréal.

1.1.3 Originally released

December 1990, C interface January 2022.

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

File Documentation

3.1 galahad_hash.h File Reference

```
#include <stdbool.h>
#include <stdint.h>
#include "galahad_precision.h"
#include "galahad_cfunctions.h"
```

Data Structures

- struct hash_control_type
- struct hash_inform_type

Functions

- void hash_initialize (void **data, struct hash_control_type *control, int *status)
- void hash_information (void **data, struct hash_inform_type *inform, int *status)
- void hash_terminate (void **data, struct hash_control_type *control, struct hash_inform_type *inform)

3.1.1 Data Structure Documentation

3.1.1.1 struct hash_control_type

Data Fields

int	error	error and warning diagnostics occur on stream error
int	out	general output occurs on stream out
int	print_level	the level of output required. Possible values are: • $\leq$ 0 no output, • $\geq$ 1 debugging
bool	space_critical	if space_critical true, every effort will be made to use as little space as possible. This may result in longer computation time
bool	deallocate_error_fatal	if deallocate_error_fatal is true, any array/pointer deallocation error will terminate execution. Otherwise, computation will continue
char	prefix[31]	all output lines will be prefixed by prefix(2:LEN(TRIM(prefix))-1) where prefix contains the required string enclosed in quotes, e.g. "string" or 'string'

3.1.1.2 struct hash_inform_type

Data Fields

int	status	return status. Possible values are: • 0 The insertion or deletion was succesful. • -1. An allocation error occurred. A message indicating the offending array is written on unit control.error, and the returned allocation status and a string containing the name of the offending array are held in inform.alloc_status and inform.bad_alloc respectively. • -2. A deallocation error occurred. A message indicating the offending array is written on unit control.error and the returned allocation status and a string containing the name of the offending array are held in inform.alloc_status and inform.bad_alloc respectively. • -99. The current dictionary is full and should be rebuilt with more space.
int	alloc_status	the status of the last attempted allocation/deallocation.
char	bad_alloc[81]	the name of the array for which an allocation/deallocation error occurred.

3.1.2 Function Documentation

3.1.2.1 hash_initialize()

```
void hash_initialize (
             void ** data,
             struct hash_control_type * control,
             int * status )
```

Set default control values and initialize private data

Parameters

in,out	*data*	holds private internal data
out	*control*	is a struct containing control information (see hash_control_type)
out	*status*	is a scalar variable of type int, that gives the exit status from the package. Possible values are (currently): • 0. The initialization was succesful.

3.1.2.2 hash_information()

```
void hash_information (
             void ** data,
```

```
            struct hash_inform_type * inform,
            int * status )
```

Provides output information

Parameters

in,out	*data*	holds private internal data
out	*inform*	is a struct containing output information (see hash_inform_type)
out	*status*	is a scalar variable of type int, that gives the exit status from the package. Possible values are (currently): • 0. The values were recorded succesfully

3.1.2.3 hash_terminate()

```
void hash_terminate (
            void ** data,
            struct hash_control_type * control,
            struct hash_inform_type * inform )
```

Deallocate all internal private storage

Parameters

in,out	*data*	holds private internal data
out	*control*	is a struct containing control information (see hash_control_type)
out	*inform*	is a struct containing output information (see hash_inform_type)