#pragma once

#include <twml/defines.h>

#include <twml/Tensor.h>

#include <twml/Type.h>

#include <stddef.h>

#ifdef \_\_cplusplus

extern "C" {

#endif

typedef void \* twml\_hashmap;

typedef int64\_t tw\_hash\_key\_t;

typedef int64\_t tw\_hash\_val\_t;

#ifdef \_\_cplusplus

}

#endif

#ifdef \_\_cplusplus

namespace twml {

typedef tw\_hash\_key\_t HashKey\_t;

typedef tw\_hash\_val\_t HashVal\_t;

class HashMap {

private:

twml\_hashmap m\_hashmap;

public:

HashMap();

~HashMap();

// Disable copy constructor and assignment

// TODO: Fix this after retain and release are added to twml\_hashmap

HashMap(const HashMap &other) = delete;

HashMap& operator=(const HashMap &other) = delete;

void clear();

uint64\_t size() const;

int8\_t insert(const HashKey\_t key);

int8\_t insert(const HashKey\_t key, const HashVal\_t val);

void remove(const HashKey\_t key);

int8\_t get(HashVal\_t &val, const HashKey\_t key) const;

void insert(Tensor &mask, const Tensor keys);

void insert(Tensor &mask, const Tensor keys, const Tensor vals);

void remove(const Tensor keys);

void get(Tensor &mask, Tensor &vals, const Tensor keys) const;

void getInplace(Tensor &mask, Tensor &keys\_vals) const;

void toTensors(Tensor &keys, Tensor &vals) const;

};

}

#endif

#ifdef \_\_cplusplus

extern "C" {

#endif

TWMLAPI twml\_err twml\_hashmap\_create(twml\_hashmap \*hashmap);

TWMLAPI twml\_err twml\_hashmap\_clear(const twml\_hashmap hashmap);

TWMLAPI twml\_err twml\_hashmap\_get\_size(uint64\_t \*size, const twml\_hashmap hashmap);

TWMLAPI twml\_err twml\_hashmap\_delete(const twml\_hashmap hashmap);

// insert, get, remove single key / value

TWMLAPI twml\_err twml\_hashmap\_insert\_key(int8\_t \*mask,

const twml\_hashmap hashmap,

const tw\_hash\_key\_t key);

TWMLAPI twml\_err twml\_hashmap\_insert\_key\_and\_value(int8\_t \*mask, twml\_hashmap hashmap,

const tw\_hash\_key\_t key,

const tw\_hash\_val\_t val);

TWMLAPI twml\_err twml\_hashmap\_remove\_key(const twml\_hashmap hashmap,

const tw\_hash\_key\_t key);

TWMLAPI twml\_err twml\_hashmap\_get\_value(int8\_t \*mask, tw\_hash\_val\_t \*val,

const twml\_hashmap hashmap,

const tw\_hash\_key\_t key);

TWMLAPI twml\_err twml\_hashmap\_insert\_keys(twml\_tensor masks,

const twml\_hashmap hashmap,

const twml\_tensor keys);

// insert, get, remove tensors of keys / values

TWMLAPI twml\_err twml\_hashmap\_insert\_keys\_and\_values(twml\_tensor masks,

twml\_hashmap hashmap,

const twml\_tensor keys,

const twml\_tensor vals);

TWMLAPI twml\_err twml\_hashmap\_remove\_keys(const twml\_hashmap hashmap,

const twml\_tensor keys);

TWMLAPI twml\_err twml\_hashmap\_get\_values(twml\_tensor masks,

twml\_tensor vals,

const twml\_hashmap hashmap,

const twml\_tensor keys);

TWMLAPI twml\_err twml\_hashmap\_get\_values\_inplace(twml\_tensor masks,

twml\_tensor keys\_vals,

const twml\_hashmap hashmap);

TWMLAPI twml\_err twml\_hashmap\_to\_tensors(twml\_tensor keys,

twml\_tensor vals,

const twml\_hashmap hashmap);

#ifdef \_\_cplusplus

}

#endif