Slice buffer design

Table of Contents

1. Overview for new design 2

1.1. Origin design 2

1.1.1. Single thread 2

1.1.2. Multi thread 3

i. 3

1.2 New design in review 4

1.2.1 Single thread 4

1.2.2. Multi-thread 5

2. Slice Buffer and thread 6

2.1 Before encoding one layer 6

2.2. Normal case for thread index and slice buffer index 7

2.3. Slice num not the same among threads 8

2.4. Different thread index in the same slice buffer 10

# Overview for new design

## Origin design

SSlice\* pSliceInLayer // slice buffer for all slices in layer

### 1.1.1. Single thread

pSliceInLayer

…

…

…

Slc3

Slc2

Slc1

Slc0

**reallocate** when current slice index larger than max slice num

…

pSliceInLayer

Slc2

Slc1

Slc0

Slc0

Slc1

Slc2

…

…

…

…

### 1.1.2. Multi thread

pSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

…

Thrd0

Thrd1

**reallocate** when current slice index larger than max slice num ,

--step 0: thread[0] detect that current slice index larger than max slice num;

--step 1: thread[0] need to wait thread[1] completed current slice encoding task

--step 2: thread[1] stop slice encoding and thread[0] reallocate slice buffer,

--step 3: thread[0]/thread[1] start to encode new slice

pSliceInLayer

Slc0

Slc1

Slc2

...

**Step 0/1**

Thrd1

Thrd0

Thrd0

pSliceInLayer

Slc0

Slc1

Slc2

…

**Step 2**

Slc0

Slc1

Slc2

...

…

…

…

**Step 3**

Thrd2

Thrd0

## 

## 1.2 New design in review

SSlice\* pSliceInLayer; //will be removed and replaced by pSliceInThread[]

SSlice\*\* ppSliceInLayer; // point to actual slice buffer

//based on slice index

SSlice\* pSliceInThread[MaxThreadNum]; // actual slice buffer

### 1.2.1 Single thread

Slc0

Slc1

Slc2

…

ppSliceInLayer

pSliceInThread[0]

Slc0

Slc1

Slc2

…

**reallocate** when current slice index larger than max slice num

Slc0

Slc1

Slc2

…

ppSliceInLayer

pSliceInThread[0]

Slc0

Slc1

Slc2

…

Slc0

Slc1

Slc2

Slc3

…

…

**reallocate**

…

Slc0

Slc1

Slc2

Slc3

…

### 1.2.2. Multi-thread

Slc0

Slc1

Slc2

Slc3

Slc4

…

ppSliceInLayer

pSliceInThread[0]

Slc4

Slc2

Slc0

…

Slc1

Slc3

Slc5

…

pSliceInThread[1]

**for reallocate**, each thread will do it independently, and will update ppSliceInLayer by ***main thread*** when all slices in layer are encoded.

ppSliceInLayer

…

Slc4

Slc3

Slc2

Slc1

Slc0

**Main thread**

Slc0

Slc1

Slc2

Slc3

Slc4

…

…

Slc4

Slc2

Slc0

…

…

Slc0

Slc2

Slc4

…

…

Slc1

Slc3

Slc5

…

…

**Thread 1**

pSliceInThread[0]

**Reallocate**

**/update**

**Reallocate**

**Thread 0**

pSliceInThread[1]

# 2. Slice Buffer and thread

## 2.1 Before encoding one layer

the status of slice buffer and thread :

example:

thread: 3 threads

slices: 9 slices in layer

bThreadBufferUsage[iThrdIdx] = false

**Slice buffer**

pSliceInThread[1]

pSliceInThread[2]

pSliceInThread[0]

Thrd\_0

**Idle**

**thread**

Thrd\_2

Thrd\_1

**Get\_UnUsed\_Buffer()**

slice in layer

Get\_Idle\_Thrd()

Slc8

Slc1

Slc2

Slc0

….

**Slice tasks**

Get\_waiting\_task()

**Task manager**

Encode\_One\_Slice()

## 2.2. Normal case for thread index and slice buffer index

**Task manager**

Thrd\_2

Slice\_2

pSliceBuffer[2]

Thrd\_1

Slice\_1

pSliceBuffer[1]

Thrd\_0

Slice\_0

pSliceBuffer[0]

Thrd\_2

Slice\_8

pSliceBuffer[2]

…

…

…

Final map for thread index and slice buffer index, slice index

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

pSliceBuffer[0]

Slc6

Slc3

Slc0

Slc1

Slc4

Slc7

pSliceBuffer[1]

Thrd\_0

Thrd\_1

Thrd\_2

pSliceBuffer[2]

Slc8

Slc5

Slc2

## 2.3. Slice num not the same among threads

case 1:

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

Slc0

Slc3

pSliceBuffer[0]

Thrd\_0

Thrd\_1

Slc7

Slc1

Slc4

pSliceBuffer[1]

Slc6

Thrd\_2

pSliceBuffer[2]

Slc8

Slc5

Slc2

case 2:

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

Slc0

pSliceBuffer[0]

Thrd\_0

Thrd\_1

Slc8

Slc1

Slc3

pSliceBuffer[1]

…

Thrd\_2

pSliceBuffer[2]

Slc2

Corner case 1, no encoded slice for one thread:

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

Slc0

Slc2

pSliceBuffer[0]

Thrd\_0

Thrd\_1

Slc7

Slc1

Slc3

pSliceBuffer[1]

…

Thrd\_2

pSliceBuffer[2]

Corner case 2, all slices encoded by one thread :

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

Thrd\_0

…

Slc1

Slc0

Slc8

pSliceBuffer[0]

Thrd\_1

pSliceBuffer[1]

Thrd\_2

pSliceBuffer[2]

## 2.4. Different thread index in the same slice buffer

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

pSliceBuffer[0]

Slc6

Slc3

Slc0

Thrd\_1

Thrd\_0

pSliceBuffer[1]

Slc7

Slc4

Slc1

Thrd\_2

pSliceBuffer[2]

Slc8

Slc5

Slc2

## 2.5. Slice index out-of-order case

ppSliceInLayer

Slc0

Slc1

Slc2

Slc3

…

Slc8

pSliceBuffer[0]

Slc2

Slc3

Slc0

Out-of-order

Thrd\_1

Thrd\_0

pSliceBuffer[1]

Slc7

Slc4

Slc1

Thrd\_2

pSliceBuffer[2]

Slc8

Slc6

Slc5

## 2.6. Dynamic slice mode case1

**Partition\_0**: **Slc0, Slc3**

Slc0

Slc3

**Partition\_1:** **Slc1, Slc4,Slc7,Slc10**

Slc1

Slc4

Slc7

Slc10

Slc5

Slc8

Slc5

Slc2

**Partition\_2:** **Slc2, Slc5, Slc8**

Slice\_Index = PartitonID + Partition\_Num \* Slice\_Index\_InPartition

Example: Slc7 = 1 + 3 \* 2

Slc5 = 2 + 3\* 1

Slice\_Index\_InLayer = PartitonOffset [Partiton\_ID]+ Slice\_Index / Partition\_Num

Here in example, PartitonOffset[0] = 0;

PartitonOffset[1] = 0 + 2 =2;

PartitonOffset[2] = 0 + 2+ 4 = 6;

Slc7 = PartitonOffset[1] + 7 / 3 = 2 + 2 = 4;

Which ppSliceInLayer[4] = Slc7

Slc8

…

Slc3

Slc2

Slc1

Slc0

ppSliceInLayer

Slc2

Slc5

Slc8

pSliceBuffer[2]

Thrd\_2

Slc7

pSliceBuffer[1]

Slc4

Slc1

Thrd\_1

Slc10

Slc0

Slc3

pSliceBuffer[0]

Thrd\_0

## 2.6. Dynamic slice mode case2

**Thread\_0 encoded two partitions**

**while no partition for Thread\_2**

**Partition\_0**: **Slc0, Slc3**

Slc0

Slc3

**Partition\_1:** **Slc1, Slc4,Slc7,Slc10**

Slc1

Slc4

Slc7

Slc10

Slc5

Slc8

Slc5

Slc2

**Partition\_2:** **Slc2, Slc5, Slc8**

Slice\_Index = PartitonID + Partition\_Num \* Slice\_Index\_InPartition

Example: Slc7 = 1 + 3 \* 2

Slc5 = 2 + 3\* 1

Slice\_Index\_InLayer = PartitonOffset [Partiton\_ID]+ Slice\_Index / Partition\_Num

Here in example, PartitonOffset[0] = 0;

PartitonOffset[1] = 0 + 2 =2;

PartitonOffset[2] = 0 + 2+ 4 = 6;

Slc7 = PartitonOffset[1] + 7 / 3 = 2 + 2 = 4;

Which ppSliceInLayer[4] = Slc7

Slc8

…

Slc3

Slc2

Slc1

Slc0

ppSliceInLayer

pSliceBuffer[2]

Slc7

pSliceBuffer[1]

Slc4

Slc1

Slc10

Thrd\_0

Slc8

Slc5

Slc2

Slc0

Slc3

pSliceBuffer[0]

Thrd\_2

Thrd\_1

3. Slice Buffer Update/Reorder

**3.1. Slice index in different slice mode.**

Dynamic slice mode

**Partition\_0**: **Slc0, Slc3**

Slc0

Slc3

**Partition\_1:** **Slc1, Slc4,Slc7,Slc10**

Slc1

Slc4

Slc7

Slc10

Slc5

Slc8

Slc5

Slc2

**Partition\_2:** **Slc2, Slc5, Slc8**

Slc0

Slc1

Slc2

Slc3

…

Slc8

ppSliceInLayer

Non-dynamic slice mode

Slc0

Slc1

…

Slc8

Slc2

Slc3