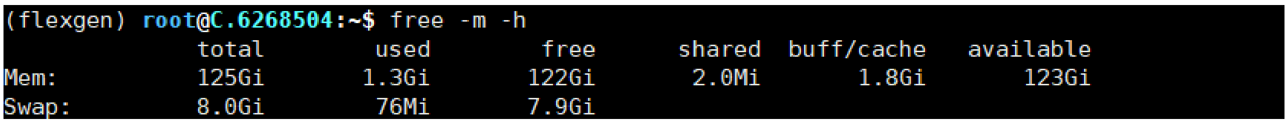
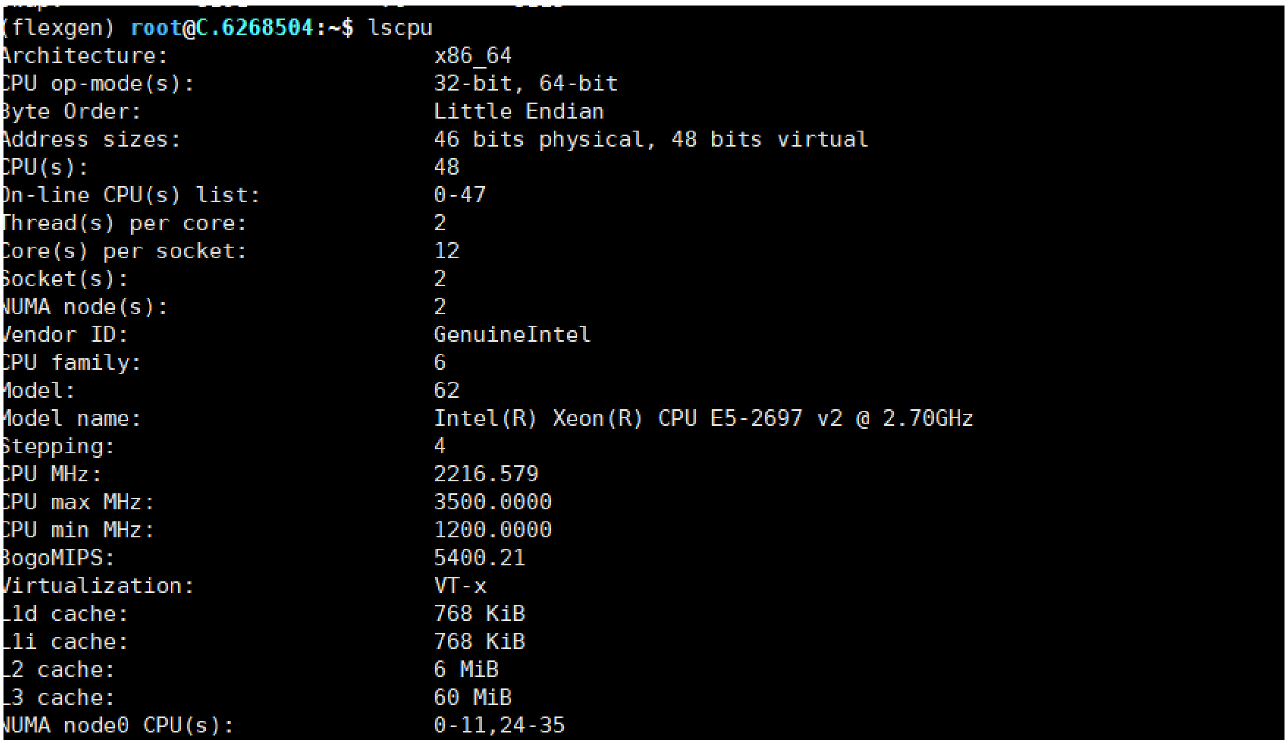
实验机器

125G 内存



```
(flexgen) root@C.6268504:~$ free -m -h
               total        used        free      shared  buff/cache   available
Mem:           125Gi       1.3Gi       122Gi       2.0Mi       1.8Gi       123Gi
Swap:          8.0Gi        76Mi       7.9Gi
```

2个CPU　12核　支持超线程　所以一共有48个逻辑CPU



```
(flexgen) root@C.6268504:~$ lscpu
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   46 bits physical, 48 bits virtual
CPU(s):                          48
On-line CPU(s) list:             0-47
Thread(s) per core:              2
Core(s) per socket:              12
Socket(s):                       2
NUMA node(s):                    2
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           62
Model name:                      Intel(R) Xeon(R) CPU E5-2697 v2 @ 2.70GHz
Stepping:                        4
CPU MHz:                         2216.579
CPU max MHz:                     3500.0000
CPU min MHz:                     1200.0000
BogoMIPS:                        5400.21
Virtualization:                  VT-x
L1d cache:                       768 KiB
L1i cache:                       768 KiB
L2 cache:                        6 MiB
L3 cache:                        60 MiB
NUMA node0 CPU(s):               0-11,24-35
```

相比之下，原论文的硬件配置如下：

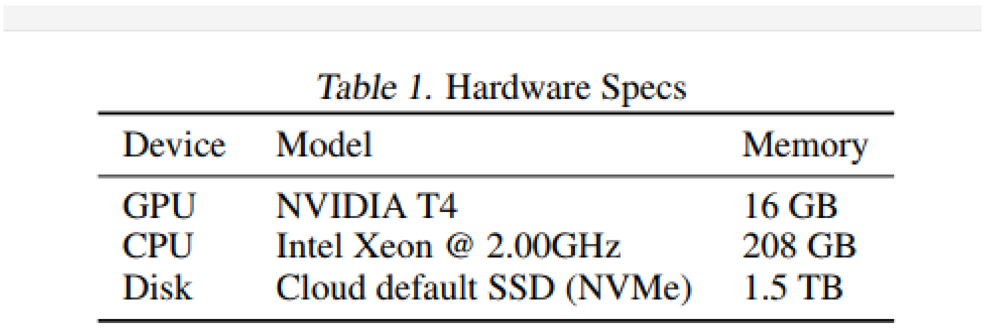


Table 1. Hardware Specs

Device	Model	Memory
GPU	NVIDIA T4	16 GB
CPU	Intel Xeon @ 2.00GHz	208 GB
Disk	Cloud default SSD (NVMe)	1.5 TB

flexgen

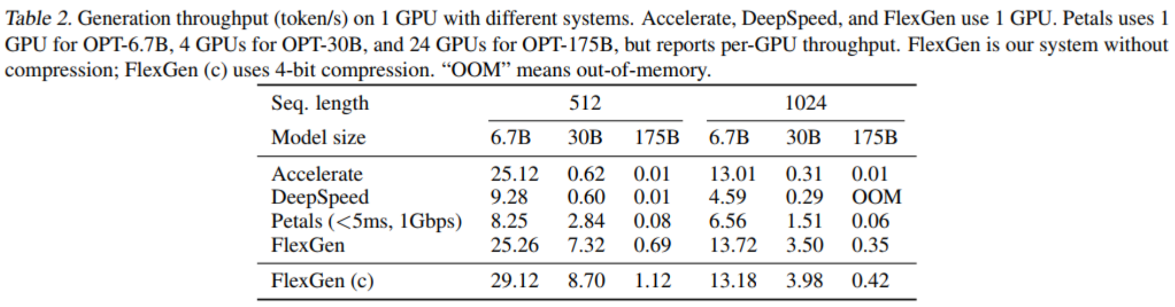


Table 2. Generation throughput (token/s) on 1 GPU with different systems. Accelerate, DeepSpeed, and FlexGen use 1 GPU. Petals uses 1 GPU for OPT-6.7B, 4 GPUs for OPT-30B, and 24 GPUs for OPT-175B, but reports per-GPU throughput. FlexGen is our system without compression; FlexGen (c) uses 4-bit compression. "OOM" means out-of-memory.

Seq. length	512			1024		
Model size	6.7B	30B	175B	6.7B	30B	175B
Accelerate	25.12	0.62	0.01	13.01	0.31	0.01
DeepSpeed	9.28	0.60	0.01	4.59	0.29	OOM
Petals (<5ms, 1Gbps)	8.25	2.84	0.08	6.56	1.51	0.06
FlexGen	25.26	7.32	0.69	13.72	3.50	0.35
FlexGen (c)	29.12	8.70	1.12	13.18	3.98	0.42

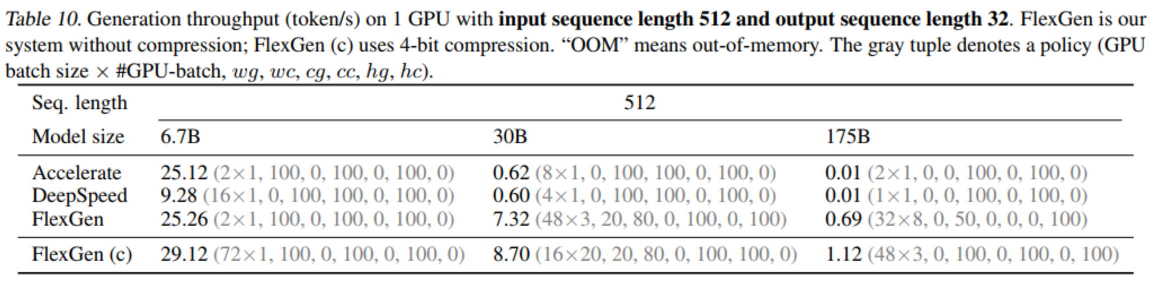


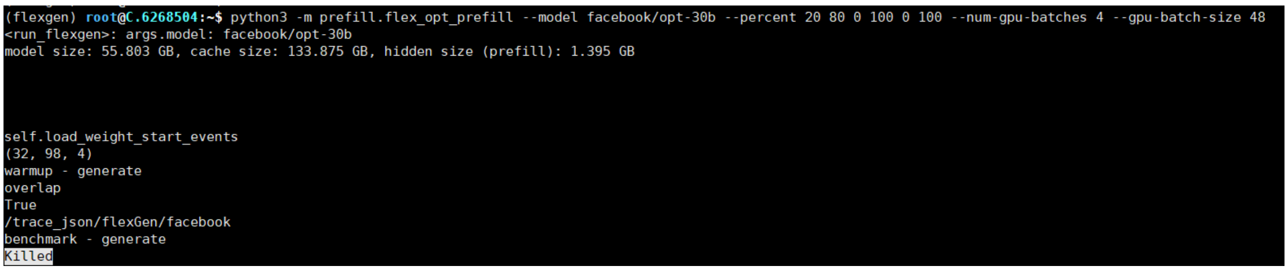
Table 10. Generation throughput (token/s) on 1 GPU with **input sequence length 512 and output sequence length 32**. FlexGen is our system without compression; FlexGen (c) uses 4-bit compression. "OOM" means out-of-memory. The gray tuple denotes a policy (GPU batch size × #GPU-batch, *wg*, *wc*, *cg*, *cc*, *hg*, *hc*).

Seq. length	512		
Model size	6.7B	30B	175B
Accelerate	25.12 (2×1, 100, 0, 100, 0, 100, 0)	0.62 (8×1, 0, 100, 100, 0, 100, 0)	0.01 (2×1, 0, 0, 100, 0, 100, 0)
DeepSpeed	9.28 (16×1, 0, 100, 100, 0, 100, 0)	0.60 (4×1, 0, 100, 100, 0, 100, 0)	0.01 (1×1, 0, 0, 100, 0, 100, 0)
FlexGen	25.26 (2×1, 100, 0, 100, 0, 100, 0)	7.32 (48×3, 20, 80, 0, 100, 0, 100)	0.69 (32×8, 0, 50, 0, 0, 0, 100)
FlexGen (c)	29.12 (72×1, 100, 0, 100, 0, 100, 0)	8.70 (16×20, 20, 80, 0, 100, 100, 0)	1.12 (48×3, 0, 100, 0, 100, 0, 100)

1 控制变量 percent 20 80 0 100 0 100 gpu-batch-size 48

1.1 opt-30b --num-gpu-batches 4

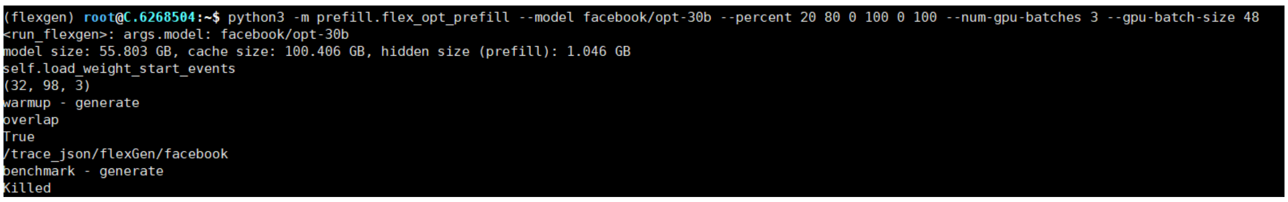
python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 4 --gpu-batch-size 48



```
(flexgen) root@C.6268504:~$ python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 4 --gpu-batch-size 48
<run_flexgen>: args.model: facebook/opt-30b
model size: 55.803 GB, cache size: 133.875 GB, hidden size (prefill): 1.395 GB




self.load_weight_start_events
(32, 98, 4)
warmup - generate
overlap
True
/trace_json/flexGen/facebook
benchmark - generate
Killed
```

本机器128GM 内存，而在benchmark generate阶段需要存储 0.8*55+133.875+1.395=179.27GB数据，超过存储空间限制，因为 内存溢出，导致被kill掉

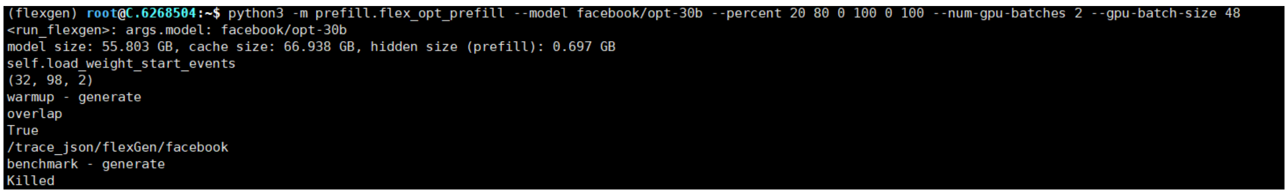
1.2 opt-30b --num-gpu-batches 3



```
(flexgen) root@C.6268504:~$ python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 3 --gpu-batch-size 48
<run_flexgen>: args.model: facebook/opt-30b
model size: 55.803 GB, cache size: 100.406 GB, hidden size (prefill): 1.046 GB
self.load_weight_start_events
(32, 98, 3)
warmup - generate
overlap
True
/trace_json/flexGen/facebook
benchmark - generate
Killed
```

本机器128GM 内存，而在benchmark generate阶段超过存储空间限制，因为 内存溢出，导致被kill掉

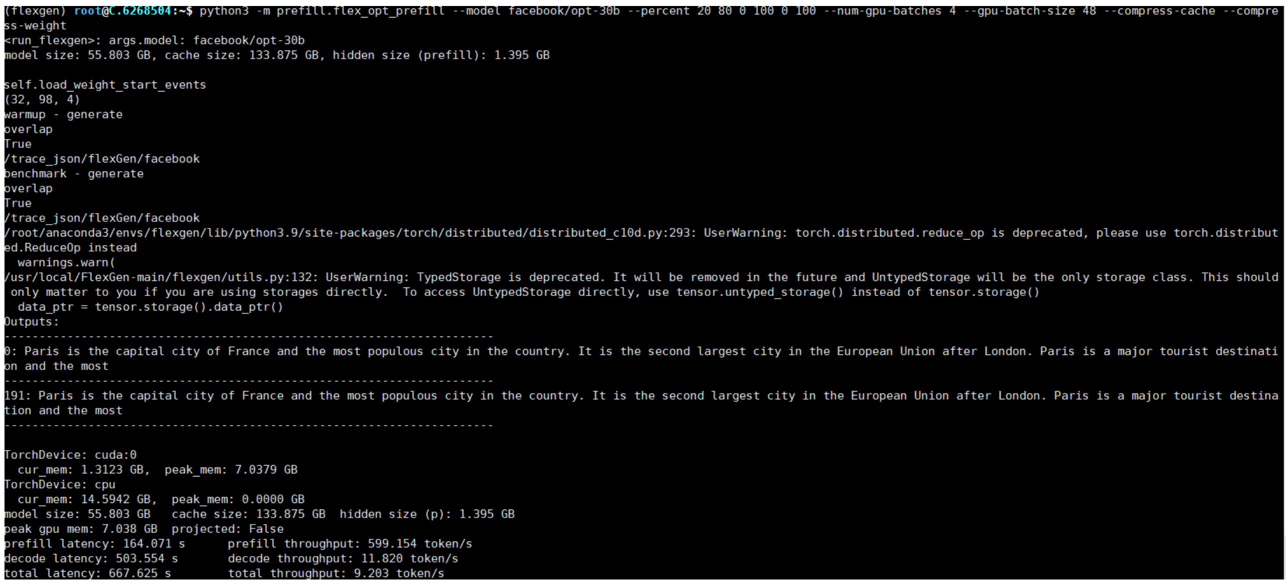
1.3 opt-30b --num-gpu-batches 2



```
(flexgen) root@C.6268504:~$ python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 2 --gpu-batch-size 48
<run_flexgen>: args.model: facebook/opt-30b
model size: 55.803 GB, cache size: 66.938 GB, hidden size (prefill): 0.697 GB
self.load_weight_start_events
(32, 98, 2)
warmup - generate
overlap
True
/trace_json/flexGen/facebook
benchmark - generate
Killed
```

本机器128GM 内存，而在benchmark generate阶段未超过存储空间限制，按理说不应该被kill

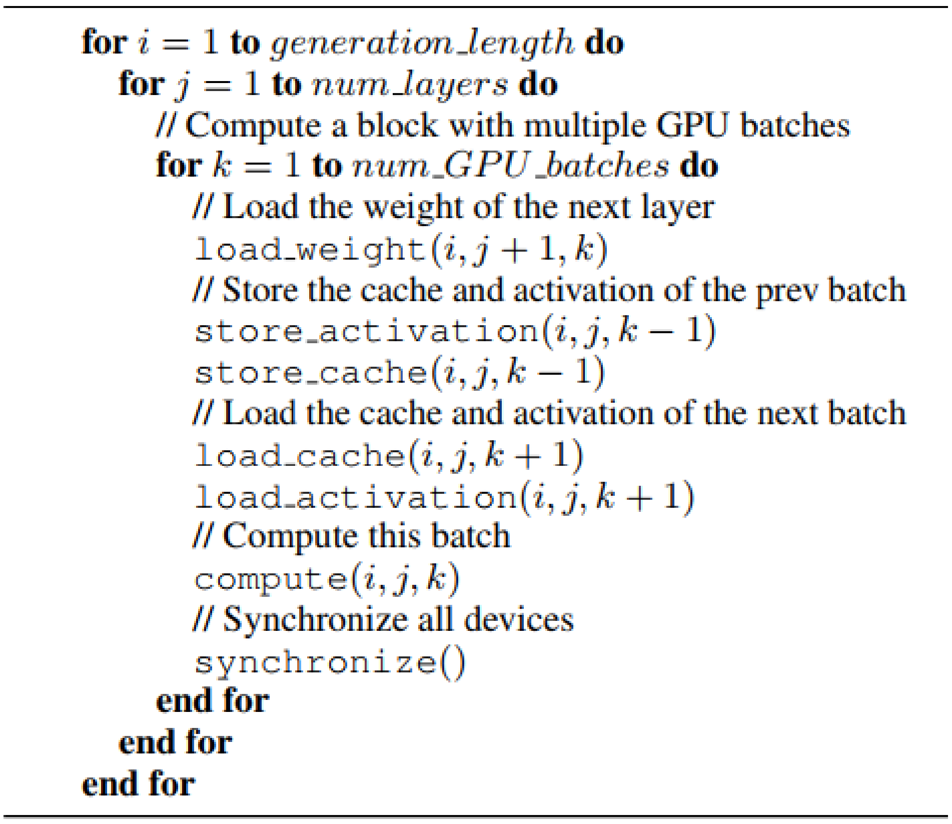
1.4 opt-30b --num-gpu-batches 4 --compress-cache --compress-weight



```
(flexgen) root@C.6268504:~$ python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 4 --gpu-batch-size 48 --compress-cache --compre
ss-weight
<run_flexgen>: args.model: facebook/opt-30b
model size: 55.803 GB, cache size: 133.875 GB, hidden size (prefill): 1.395 GB

self.load_weight_start_events
(32, 98, 4)
warmup - generate
overlap
True
/trace_json/flexGen/facebook
benchmark - generate
overlap
True
/trace_json/flexGen/facebook
/root/anaconda3/envs/flexgen/lib/python3.9/site-packages/torch/distributed/distributed_c10d.py:293: UserWarning: torch.distributed.reduce_op is deprecated, please use torch.distribut
ed.ReduceOp instead
  warnings.warn(
/usr/local/FlexGen-main/flexgen/utils.py:132: UserWarning: TypedStorage is deprecated. It will be removed in the future and UntypedStorage will be the only storage class. This should
 only matter to you if you are using storages directly.  To access UntypedStorage directly, use tensor.untyped_storage() instead of tensor.storage()
  data_ptr = tensor.storage().data_ptr()
Outputs:
----------------------------------------------------------------------
0: Paris is the capital city of France and the most populous city in the country. It is the second largest city in the European Union after London. Paris is a major tourist destinati
on and the most
----------------------------------------------------------------------
191: Paris is the capital city of France and the most populous city in the country. It is the second largest city in the European Union after London. Paris is a major tourist destina
tion and the most
----------------------------------------------------------------------

TorchDevice: cuda:0
  cur_mem: 1.3123 GB,  peak_mem: 7.0379 GB
TorchDevice: cpu
  cur_mem: 14.5942 GB,  peak_mem: 0.0000 GB
model size: 55.803 GB   cache size: 133.875 GB  hidden size (p): 1.395 GB
peak gpu mem: 7.038 GB  projected: False
prefill latency: 164.071 s      prefill throughput: 599.154 token/s
decode latency: 503.554 s       decode throughput: 11.820 token/s
total latency: 667.625 s        total throughput: 9.203 token/s
```

和原文结果接近，牛!

1.4.1 profile文件分析

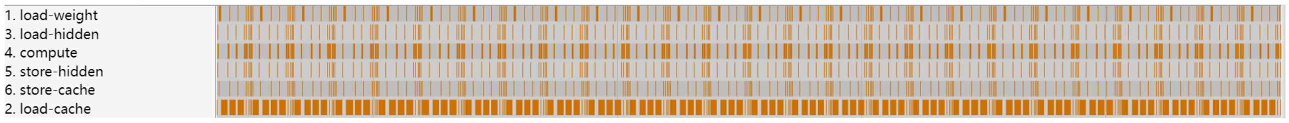


```
for $i = 1$ to $generation\_length$ do
    for $j = 1$ to $num\_layers$ do
        // Compute a block with multiple GPU batches
        for $k = 1$ to $num\_GPU\_batches$ do
            // Load the weight of the next layer
            load_weight($i, j + 1, k$)
            // Store the cache and activation of the prev batch
            store_activation($i, j, k - 1$)
            store_cache($i, j, k - 1$)
            // Load the cache and activation of the next batch
            load_cache($i, j, k + 1$)
            load_activation($i, j, k + 1$)
            // Compute this batch
            compute($i, j, k$)
            // Synchronize all devices
            synchronize()
        end for
    end for
end for
```

Figure 4. Block schedule with overlapping.

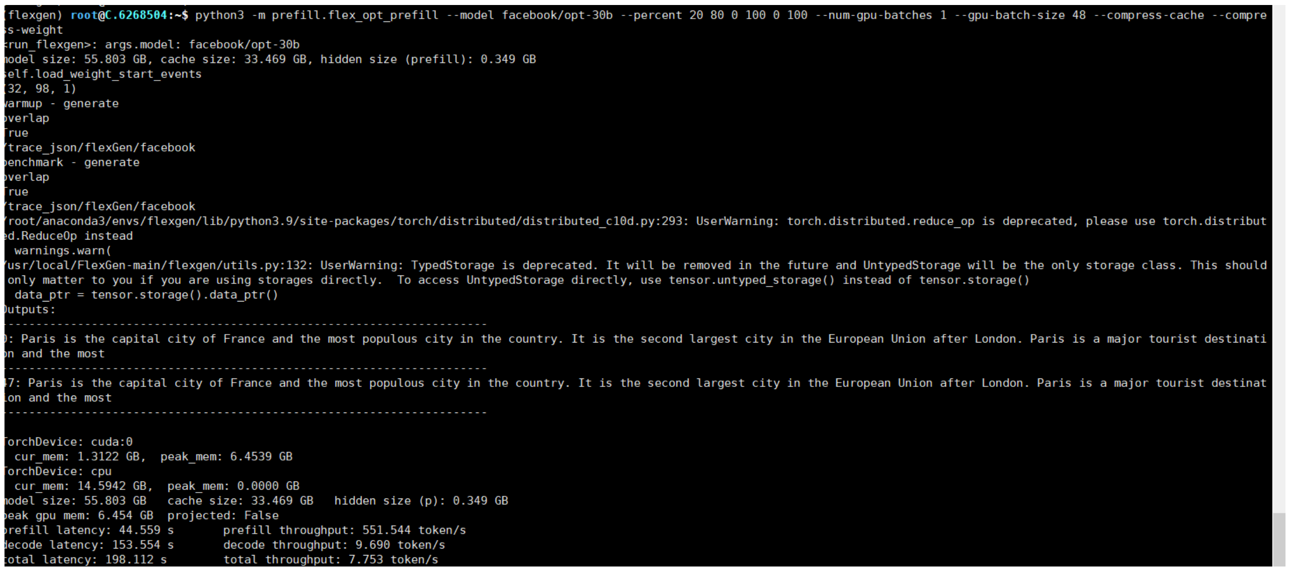
- 各个步骤的顺序满足上述伪代码表述的顺序

- **问题：各个步骤并没有时间上的重叠?**
- prefill阶段后，各个步骤的前后间隔呈现很强的规律性



- 在prefill阶段过后，即gen_len_id大于0时，有了load_cache和store_cache操作，compute的时间显著降低，证明overlap是有用滴。此时性能瓶颈从compute变成了load_cache。之后基本都是这样的pattern

1.5 opt-30b num-gpu-batches 1 --compress-cache --compress-weight



```
(flexgen) root@C.6268504:~$ python3 -m prefill.flex_opt_prefill --model facebook/opt-30b --percent 20 80 0 100 0 100 --num-gpu-batches 1 --gpu-batch-size 48 --compress-cache --compre
ss-weight
<run_flexgen>: args.model: facebook/opt-30b
nodel size: 55.803 GB, cache size: 33.469 GB, hidden size (prefill): 0.349 GB
self.load_weight_start_events
(32, 98, 1)
warmup - generate
overlap
True
/trace_json/flexGen/facebook
benchmark - generate
overlap
True
/trace_json/flexGen/facebook
/root/anaconda3/envs/flexgen/lib/python3.9/site-packages/torch/distributed/distributed_c10d.py:293: UserWarning: torch.distributed.reduce_op is deprecated, please use torch.distribut
ed.ReduceOp instead
  warnings.warn(
/usr/local/FlexGen-main/flexgen/utils.py:132: UserWarning: TypedStorage is deprecated. It will be removed in the future and UntypedStorage will be the only storage class. This should
 only matter to you if you are using storages directly.  To access UntypedStorage directly, use tensor.untyped_storage() instead of tensor.storage()
  data_ptr = tensor.storage().data_ptr()
Outputs:
----------------------------------------------------------------------
0: Paris is the capital city of France and the most populous city in the country. It is the second largest city in the European Union after London. Paris is a major tourist destinati
on and the most
----------------------------------------------------------------------
47: Paris is the capital city of France and the most populous city in the country. It is the second largest city in the European Union after London. Paris is a major tourist destinat
ion and the most
----------------------------------------------------------------------

TorchDevice: cuda:0
  cur_mem: 1.3122 GB,  peak_mem: 6.4539 GB
TorchDevice: cpu
  cur_mem: 14.5942 GB,  peak_mem: 0.0000 GB
model size: 55.803 GB   cache size: 33.469 GB   hidden size (p): 0.349 GB
peak gpu mem: 6.454 GB  projected: False
prefill latency: 44.559 s       prefill throughput: 551.544 token/s
decode latency: 153.554 s       decode throughput: 9.690 token/s
total latency: 198.112 s        total throughput: 7.753 token/s
```

和原文结果接近，it makes sense

2 facebook/opt-175b --percent 0 0 100 0 100 0

2.1 opt-175b 需要申请获得权重参数

```
python3 -m flexgen.flex_opt_prefill --model facebook/opt-175b --percent 0 0 100 0 100 0 --num-gpu-batches 4 --offload-dir /user/local/flexgen/offload
```

dist-flexgen

Table 3. The scaling performance on 4 GPUs. The prompt sequence length is 512. Generation throughput (token/s) counts the time cost of both prefill and decoding while decoding throughput only counts the time cost of decoding assuming prefill is done.

Metric	Generation Throughput			Decoding Throughput		
Model size	6.7B	30B	175B	6.7B	30B	175B
FlexGen 1 GPU	25.26	7.32	0.69	38.28	11.52	0.83
FlexGen 4 GPU	201.12	23.61	2.33	764.65	48.94	3.86
DeepSpeed 4 GPU	50.00	6.40	0.05	50.20	6.40	0.05

need to be compressed

1 控制变量 percent 20 80 0 100 0 100 gpu-batch-size 12

1.1 opt-30b num-gpu-batches 3 --compress-weight --compress-cache

mpirun --allow-run-as-root --mca btl_tcp_if_exclude lo,docker0 --mca oob_tcp_if_exclude lo,docker0 --map-by ppr:4:node:pe=12 --oversubscribe -H 172.17.0.3 --bind-to core:overload-allowed -x OMP_NUM_THREADS=12 --mca orte_base_help_aggregate 0 --verbose /root/anaconda3/envs/flexgen/bin/python -m prefill.dist_flex_opt_prefill --head-ip 172.17.0.3 --port 7777 --use-mpi --model facebook/opt-30b --gpu-batch-size 12 --num-gpu-batches 3 --percent 20 80 0 100 0 100 --comm-device cpu --compress-weight --compress-cache --path *DUMMY* --cut-gen-len 2 --cpu

```
WARNING: Open MPI tried to bind a process but failed.  This is a
warning only; your job will continue, though performance may
be degraded.

  Local host:        6a251bf104a5
  Application name:  /root/anaconda3/envs/flexgen/bin/python
  Error message:     failed to bind memory
  Location:          rtc_hwloc.c:447
```

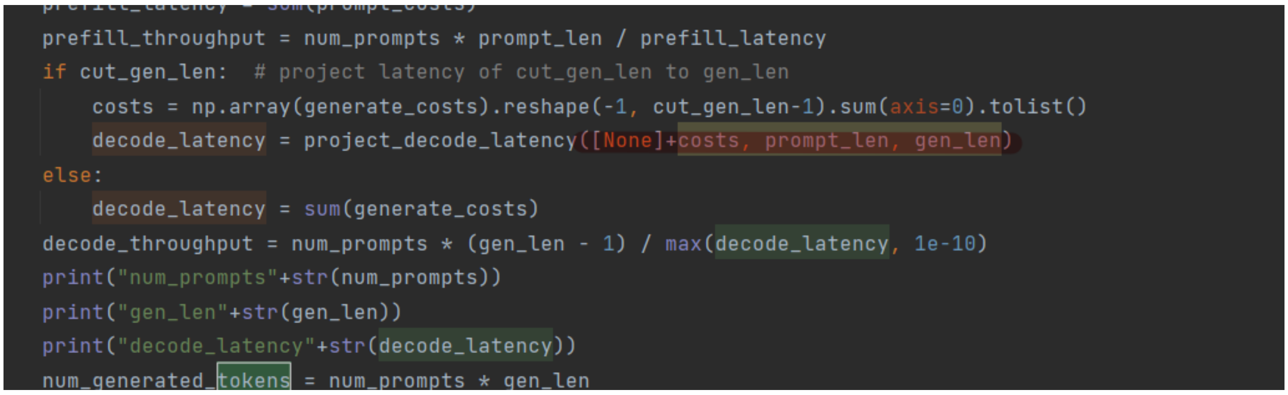
```
WARNING: Open MPI tried to bind a process but failed.  This is a
warning only; your job will continue, though performance may
be degraded.

  Local host:        6a251bf104a5
  Application name:  /root/anaconda3/envs/flexgen/bin/python
  Error message:     failed to bind memory
  Location:          rtc_hwloc.c:447
```

```
WARNING: Open MPI tried to bind a process but failed.  This is a
warning only; your job will continue, though performance may
be degraded.

  Local host:        6a251bf104a5
  Application name:  /root/anaconda3/envs/flexgen/bin/python
  Error message:     failed to bind memory
  Location:          rtc_hwloc.c:447
```

```
WARNING: Open MPI tried to bind a process but failed.  This is a
warning only; your job will continue, though performance may
be degraded.

  Local host:        6a251bf104a5
  Application name:  /root/anaconda3/envs/flexgen/bin/python
  Error message:     failed to bind memory
  Location:          rtc_hwloc.c:447
```

```
test
Initializing distributed environment at 172.17.0.3:7777, world_size=4, rank=0, local_rank=0.
test
Initializing distributed environment at 172.17.0.3:7777, world_size=4, rank=1, local_rank=1.
test
Initializing distributed environment at 172.17.0.3:7777, world_size=4, rank=2, local_rank=2.
test
Initializing distributed environment at 172.17.0.3:7777, world_size=4, rank=3, local_rank=3.
rank #3: Finished initializing distributed environment
rank #1: Finished initializing distributed environment
rank #0: Finished initializing distributed environment
rank #2: Finished initializing distributed environment
rank #0: global_rank0
rank #3: global_rank3
rank #2: global_rank2
rank #1: global_rank1
rank #2: =Current micro-batch send/recv size: 1344 MB (fp16)
rank #2: ===Number of micro-batches: 4.
rank #2: model size: 55.803 GB, cache size: 100.406 GB, hidden size (prefill): 1.046 GB
rank #2: warmup - generate
rank #1: ===Current micro-batch send/recv size: 1344 MB (fp16)
rank #1: ===Number of micro-batches: 4.
rank #1: model size: 55.803 GB, cache size: 100.406 GB, hidden size (prefill): 1.046 GB
rank #1: warmup - generate
rank #0: ===Current micro-batch send/recv size: 1344 MB (fp16)
rank #0: ===Number of micro-batches: 4.
rank #0: model size: 55.803 GB, cache size: 100.406 GB, hidden size (prefill): 1.046 GB
rank #0: warmup - generate
rank #3: ===Current micro-batch send/recv size: 1344 MB (fp16)
rank #3: =====Number of micro-batches: 4.
rank #3: model size: 55.803 GB, cache size: 100.406 GB, hidden size (prefill): 1.046 GB
rank #3: warmup - generate
rank #3: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:3.json
rank #1: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:1.json
rank #2: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:2.json
rank #0: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:0.json
rank #1: benchmark - generate
rank #3: benchmark - generate
rank #2: benchmark - generate
rank #0: benchmark - generate
rank #3: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:3.json
rank #1: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:1.json
rank #2: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:2.json
rank #0: output:prefilling/generate_overlap_Truenum_gpu_batches_3_percent[20, 80, 0, 100, 0,
100]pp_rank:0.json
/root/anaconda3/envs/flexgen/lib/python3.9/site-packages/numpy/core/fromnumeric.py:3464: RuntimeWarning:
Mean of empty slice.
  return methods.mean(a, axis=axis, dtype=dtype,
/root/anaconda3/envs/flexgen/lib/python3.9/site-packages/numpy/core/_methods.py:192: RuntimeWarning:
invalid value encountered in scalar divide
  ret = ret.dtype.type(ret / rcount)
/root/anaconda3/envs/flexgen/lib/python3.9/site-packages/torch/distributed/distributed_c10d.py:293:
UserWarning: torch.distributed.reduce_op is deprecated, please use torch.distributed.ReduceOp instead
  warnings.warn(
/usr/local/FlexGen-main/flexgen/utils.py:132: UserWarning: TypedStorage is deprecated. It will be removed in the
future and UntypedStorage will be the only storage class. This should only matter to you if you are using storages
directly.  To access UntypedStorage directly, use tensor.untyped_storage() instead of tensor.storage()
  data_ptr = tensor.storage().data_ptr()
rank #3: TorchDevice: cuda:3
rank #3:   cur_mem: 0.3319 GB,  peak_mem: 2.1732 GB
rank #3: TorchDevice: cpu
rank #3:   cur_mem: 4.0911 GB,  peak_mem: 0.0000 GB
rank #3: model size: 55.803 GB  cache size: 100.406 GB  hidden size (prefill): 1.046 GB
peak gpu mem: 2.173 GB
prefill latency: 68.99 s    prefill throughput: 1068.64 token/s
decode latency: nan s    decode throughput: nan token/s
total latency: nan s   total throughput: nan token/s
```

1.2 源代码似乎吧decode lantency给隐藏起来了



```python
prefill_latency = sum(prompt_costs)
prefill_throughput = num_prompts * prompt_len / prefill_latency
if cut_gen_len:  # project latency of cut_gen_len to gen_len
    costs = np.array(generate_costs).reshape(-1, cut_gen_len-1).sum(axis=0).tolist()
    decode_latency = project_decode_latency([None]+costs, prompt_len, gen_len)
else:
    decode_latency = sum(generate_costs)
decode_throughput = num_prompts * (gen_len - 1) / max(decode_latency, 1e-10)
print("num_prompts"+str(num_prompts))
print("gen_len"+str(gen_len))
print("decode_latency"+str(decode_latency))
num_generated_tokens = num_prompts * gen_len
```

costs不为nan 但是decode_latency为nan