



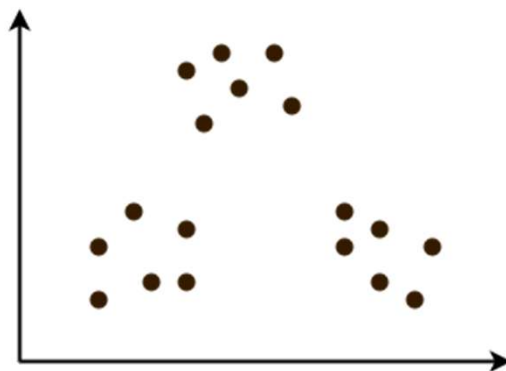
Parallel k-means clustering

김민수

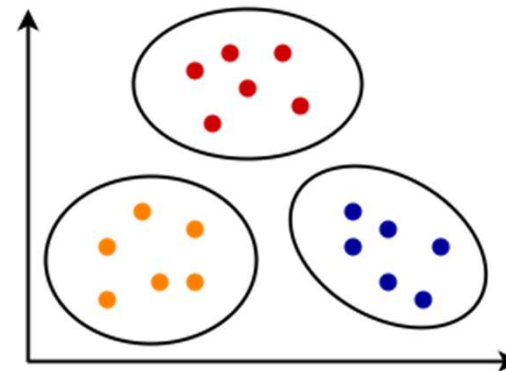
2025-22681

K-means clustering

- K-means는 각 데이터 포인트를 가장 가까운 중심점 에 할당
- 반복적 최적화
 - Assignment step: 데이터 → 가장 가까운 centroid로 배정
 - Update step: centroid 재계산 (거리 평균)
 - 반복
- 거리 계산 → 병렬화에 매우 적합



Before K-Means



After K-Means

Dataset

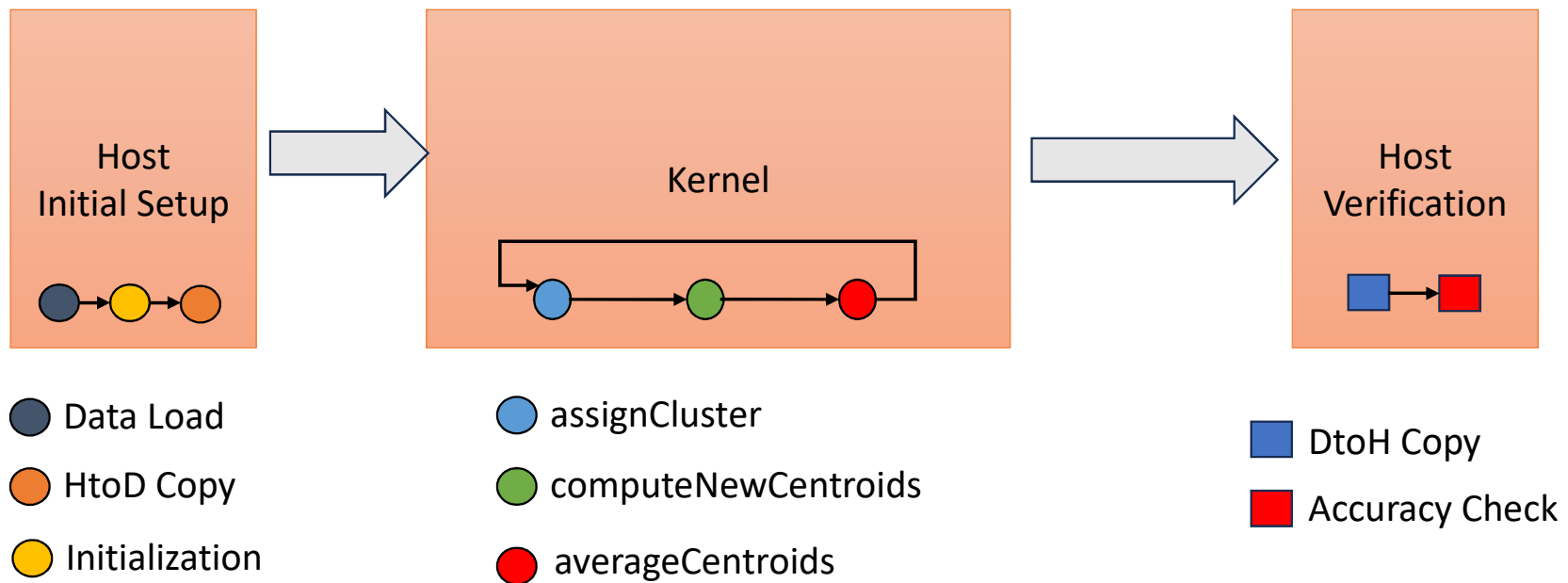
- make_blobs 기반 K-means 적합 synthetic 데이터
- $N=10M$, $\text{dim} = 100$, # of Cluster: 10
- **Dense / Sparse로 랜덤 구성**
 - Dense: sparsity 0–20% Sparse: sparsity 80–95%
- Average로 성능 측정.
- First cluster centroids은 맨 앞 10개 points

BaseLine

- CPU Baseline – sklearn
 - Python 기반의 대표적인 머신러닝 라이브러리
 - CPU 환경에서 K-means를 안정적으로 구현, 계산 병목을 명확히 관찰 가능
 - Average : 24556 ms
- GPU Baseline – cuML
 - NVIDIA RAPIDS 기반의 GPU 가속 머신러닝 라이브러리
 - CUDA 기반 병렬 연산을 통해 K-means를 고속 수행
 - Average : 993ms

Flow - Naïve

- Naïve imple.
 - Average : 3211.6382ms
 - Precision 손실 double 타입 보정



Key design ideas

- Memory Hierarchy Maximization
 - 메모리 특성에 맞춰 최적화
 - Shared Memory와 Constant Memory 사용
 - Coalesced access을 위해 SOA 사용
 - Sparsity > 80% 처리 시 압축을 통해 연산 효율 증대

Implementation

1. Constant Memory 사용 - Assignment step

- Assignment 에서 사용되는 Centroids 데이터는 Read-Only.
- 각 iteration 마다 GPU의 **Constant Memory (__constant__)** 영역에 할당.

- `__constant__ float c_centroids[N_CLUSTERS * MAX_FEATURES];`
- `CUDA_CHECK(cudaMemcpyToSymbol(c_centroids, d_centroids, centroid_size_float));`

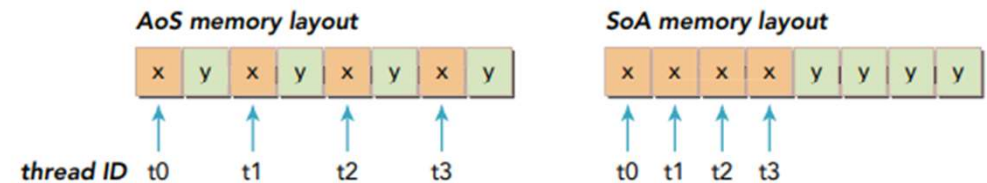
2. Shared Memory Reduction - Compute New Centroids step

- 각 point가 속하는 cluster에 관한 정보를 블록 내 Shared Memory에 우선 누적 후 한번에 add
- Contention 최소화

Implementation

3. AoS -> SoA - Assignment step

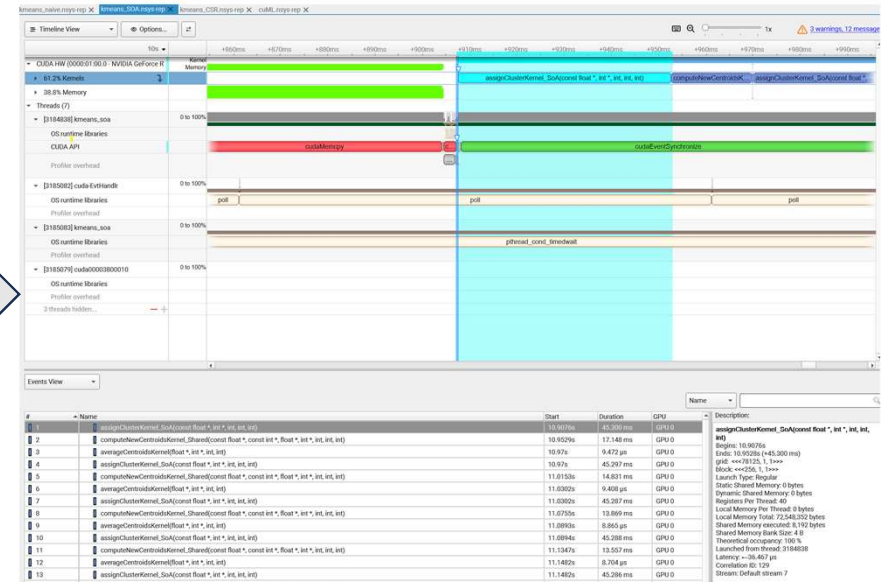
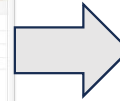
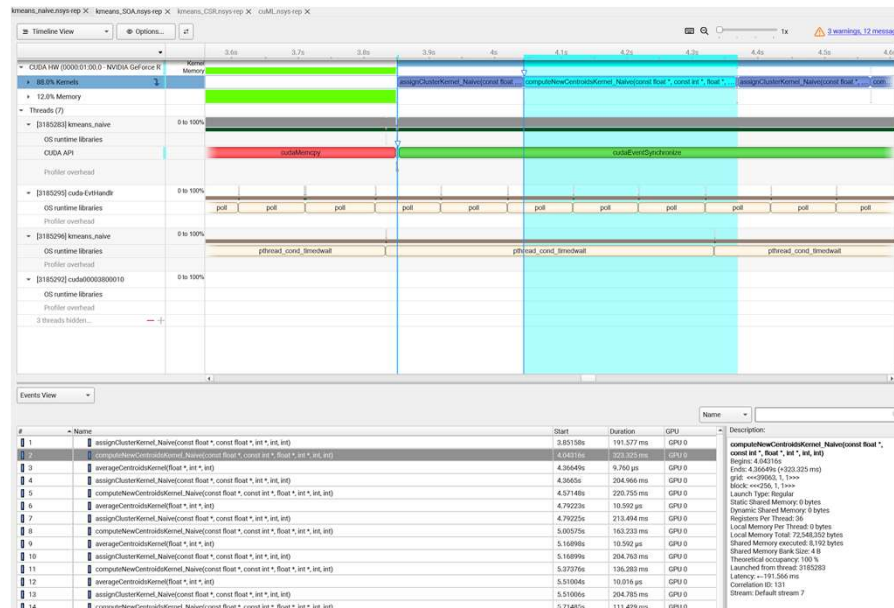
- AOS : 데이터가 [샘플 1 전체], [샘플 2 전체], ... 순서로 저장.
 - (예: [x1, y1, z1, x2, y2, z2, ...])
- SoA : 데이터가 [feature1 배열], [feature2 배열], ...
 - 예: [x1, x2, ...], [y1, y2, ...], [z1, z2, ...]
 - 각 Thread가 하나의 포인트를 담당하고 Thread들은 x->y->z..처럼 순차적으로 centroid와의 거리 계산.



Implementation

3. 1~3적용시 Naïve 대비 개선

- assignClusterKernel **191.5 ms -> 45.3 ms**
- computeNewCentroidsKernel **323.3 ms-> 17.1ms**



Implementation

4. CSR 적용

- Data가 sparse하면 csr format으로 host에서 변경 후 연산.
- 0인 값들에 대한 연산횟수 감소 및 효율적 연산 가능

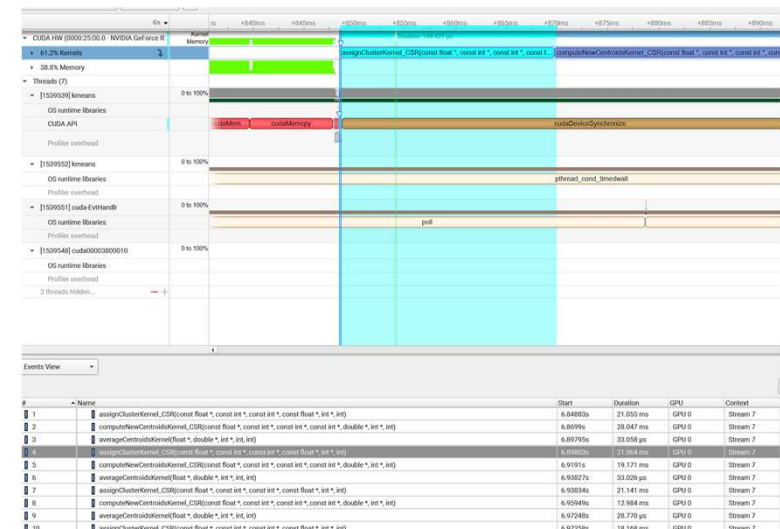
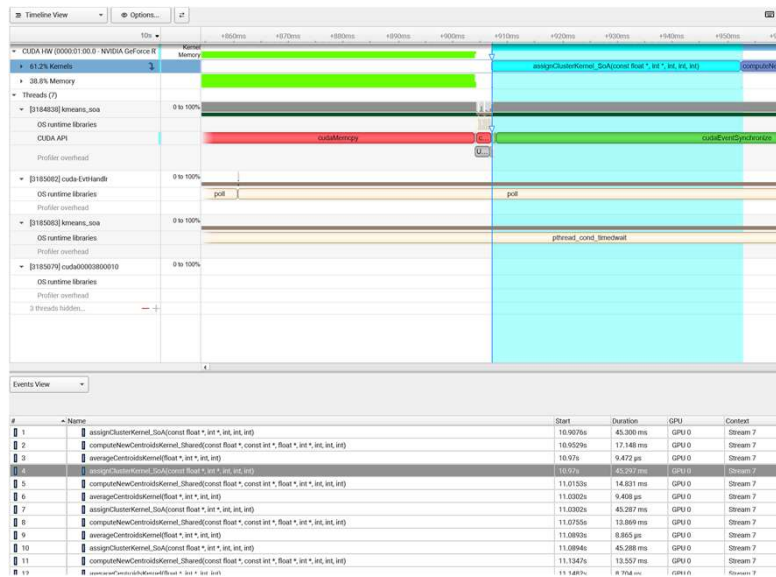
$$D(\mathbf{x}, \mathbf{c}) = \sum_{f=1}^M (x_f - c_f)^2 \quad \longrightarrow \quad D(\mathbf{x}, \mathbf{c}) = \underbrace{\sum x_f^2}_{\text{I}} + \underbrace{\sum c_f^2}_{\text{II}} - \underbrace{2 \sum x_f c_f}_{\text{III}}$$

- 1,2번 항은 한번씩만 계산 후 활용 .
- Data에서 feature 값이 0인경우는 3번항에서 계산할 필요가없다.
- But Dense의 경우 오히려 느려짐.

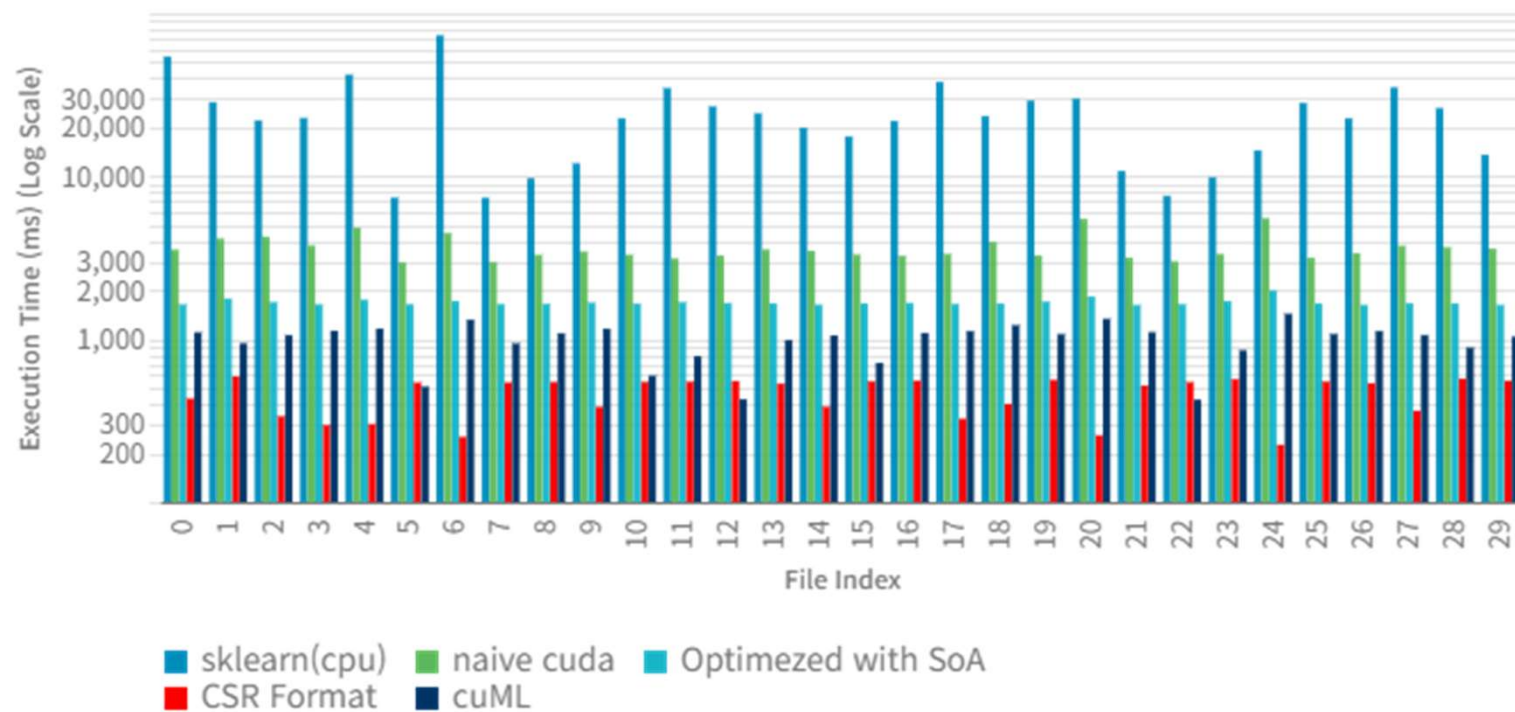
Implementation

4. CSR 적용

- 전후 assignClusterKernel 1회 연산 시간 **45.3ms -> 21.0ms**



Experiment result



Experiment result

- CSR 적용시 가장 좋은 결과 달성

Implementation	Average Execution Time (ms)	Speedup vs CPU
CSR Format	462.45 ms	53.10x
cuML	993.25 ms	24.72x
Optimezed with SoA	1679.87 ms	14.62x
naive cuda	3650.78 ms	6.73x
sklearn(cpu)	24556.43 ms	1.00x

Limitation

- CSR 적용을 위한 추가적인 연산 시간

항목	평균 시간 (ms)	상세 분석
Sparsity Analysis Time	346 ms	Host- Sparsity 계산에 소요.
CSR Conversion Time	4192 ms	Host - CSR Mode 일 때만 발생. Dense 포함 평균 2515ms
Execution Time (CUDA)	438 ms	Kernel - 순수 GPU 컴퓨팅 시간.

Conclusion

- 데이터 특성을 통한 SoA, CSR format이 가속에 매우 효과적이었음.
- 전처리가 이미 되어있다면 CSR을 적용하는 것이 cuML 사용보다 우위
- CUDA를 통해 직접 가속화를 하면 data의 특성을 반영하기 좋고 flexibility가 높아짐.

Reference Reproduce

Popcorn : Accelerating Kernel K-means on GPUs through Sparse Linear Algebra

Popcorn Reproduce

- Abstract
 - Kernel K-means 최적화
 - 데이터를 비선형 함수 $\phi(x)$ 로 고차원 공간에 보내 복잡한 형태의 군집도 잘 분리
 - 이때 거리 행렬 D (각 점과 각 centroid 사이의 제곱거리)는 다음으로 쓸 수 있다. $D = -2KVT + P\sim + C\sim$
 - K 커널행렬, V 선택/정규화 된 희소행렬
 - $P\sim$ 각점의 각 점의 $\|\phi(p_i)\|^2$ 를 복제한 행렬
 - $C\sim$ 는 centroids의 노름 $\|c_j\|^2$ 를 복제한 행렬

Popcorn Reproduce

- What to do to improve upon it?
 - **Tensor Core 적극 활용하기**
 1. **fp32를 fp16**으로 변환, exp, mentisa 를 전용 함수를 사용하여 변환
 2. cublas에서 호출되는 함수를 FP16의 tensor core를 사용하는 버전으로 호출되게 변경
 3. 커스텀 커널들도 필요에 따라서 fp16 버전으로 컨버전
 - **BSR, RCM 알고리즘으로 재정렬을 통해 locality 증가** 시키기.

Popcorn Reproduce

- Build 관련 문제
 - Dependency 설명 x, fmt 와 다른 lib간 충돌로 인한 빌드 오류, Cmake Build time 2시간 이상

```
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/fmt-src/include/fmt/format-inl.h:150:7: error: expected ';' before 'class'
150 |     return std::system_error(ec, vformat(fmt, args));
    |           ^
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/fmt-src/include/fmt/format-inl.h:150:8: error: expected primary-expression before 'class'
150 |     return std::system_error(ec, vformat(fmt, args));
    |           ^
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/fmt-src/include/fmt/format-inl.h: In function 'void fmt::v10::format_system_error(fmt::v10::detail::basic_format_context::iterator, int, const char*)':
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/fmt-src/include/fmt/format-inl.h:1408:32: error: expected primary-expression before 'class'
1408 |     write(std::back_inserter(out), std::system_error(ec, message).what());
    |                                ^
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/raft-src/cpp/include/raft/linalg/gemm.cuh: At global scope:
/home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/raft-src/cpp/include/raft/linalg/gemm.cuh:22:228: note: #pragma message: /home/s7/minsu/workspace/computing2/Project/popcorn/Popcorn/build2/_deps/raft-src/cpp/include/raft/linalg/gemm.cuh is deprecated and will be removed in a future release. Use raft/linalg/gemm.hpp instead.
22 | #pragma message(__FILE__
    |
```

```
minsu@login0:~/workspace/computing2/Project/popcorn/Popcorn/build2$ cmake --build . --target gpukmeans -j16 2>&1 | tee build_log2.txt
[ 0%] Building CUDA object CMakeFiles/raft-build/CMakeFiles/raft_objs.dir/src/cuda_utils.cu.o
[ 0%] Building CXX object _deps/raft-build/CMakeFiles/raft_objs.dir/src/core/logger.cpp.o
[ 0%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_canberra_double_double_double_int.cu.o
[ 2%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_correlation_double_double_double_int.cu.o
[ 2%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_correlation_float_float_float_int.cu.o
[ 3%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_dice_double_double_double_int.cu.o
[ 3%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_cosine_float_float_float_int.cu.o
[ 3%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_hamming_unexpanded_double_double_double_int.cu.o
[ 3%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_hamming_unexpanded_float_float_float_int.cu.o
[ 5%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_jensen_shannon_double_double_double_int.cu.o
[ 5%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_cosine_double_double_double_int.cu.o
[ 4%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_hellinger_expanded_float_float_float_int.cu.o
[ 5%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_dice_float_float_float_int.cu.o
[ 5%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_canberra_float_float_float_int.cu.o
[ 5%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_jensen_shannon_float_float_float_int.cu.o
[ 6%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_hellinger_expanded_double_double_double_int.cu.o
[ 6%] Linking CUDA static library libutils.a
[ 6%] Building target utils
[ 6%] Building CUDA object _deps/raft-build/CMakeFiles/raft_objs.dir/src/distance/detail/pairwise_matrix/dispatch_kl_divergence_double_double_double_int.cu.o
```

Popcorn Reproduce

- Reproduce 실패.
 - Naïve 버전
 - 제공된 테스트 케이스(tc1, tc2)가 재현되지 않음
 - tc1 실행은 되지만 **accuracy 부정확**.
 - tc2 1개의 gpu에서 실행 안됨.
 - Popcorn 버전
 - 공개된 코드에서 메모리 관리가 제대로 되지 않아 fail.
 - 디버깅 시도했으나 상당한 시간 소요.
 - 일부 코드가 완전히 구현되지 않은 것으로 보이며, 이로 인해 재현이 어려움.

```
[PERFORMANCE] compute_centroids time: 0.00003072
[PERFORMANCE] compute_distances time: 0.00008909
[PERFORMANCE] clusters_argmin_shfl time: 0.00002458
[PERFORMANCE] compute_centroids time: 0.00003072
[PERFORMANCE] compute_distances time: 0.00008912
[PERFORMANCE] clusters_argmin_shfl time: 0.00002560
[PERFORMANCE] compute_centroids time: 0.00003178
[PERFORMANCE] compute_distances time: 0.00008806
[PERFORMANCE] clusters_argmin_shfl time: 0.00002560
[PERFORMANCE] compute_centroids time: 0.00007578
[PERFORMANCE] compute_distances time: 0.00008899
[PERFORMANCE] clusters_argmin_shfl time: 0.00002458
[PERFORMANCE] compute_centroids time: 0.00003174
[PERFORMANCE] compute_distances time: 0.00008909
[PERFORMANCE] clusters_argmin_shfl time: 0.00002464
[PERFORMANCE] compute_centroids time: 0.00003174
[PERFORMANCE] compute_distances time: 0.00008806
[PERFORMANCE] clusters_argmin_shfl time: 0.00002458
[PERFORMANCE] compute_centroids time: 0.00003082
[PERFORMANCE] compute_distances time: 0.00008704
[PERFORMANCE] clusters_argmin_shfl time: 0.00002461
[PERFORMANCE] compute_centroids time: 0.00003174
MEMORY FOOTPRINT: 299 MB
K-means did NOT converge - Time: 0.066605
Objective score: -5794003.000000
double free or corruption (out)
Aborted (core dumped)
```



End