We integrate key encoding and rearrange into multiway merge operation
- multiway merge, which merges \( k \) \((k > 2)\) input streams into one output stream, is a common technique to reduce memory bandwidth in mergesort

Steps of our multiway merge operation
1. at the first stage, read records from system memory and encode key and streamID into an integer
2. merge integer values using SIMD
3. at the last stage, rearrange records based on the encoded streamID

Optimizations and overall scheme

- **Key idea:**
  - to execute rearranging of records more frequently, e.g. once per \( m \) merge stages \((m > 1)\), instead of only once at the last in key-index approach

**Benefits:**
- Cache friendly: the rearrange operation reads from \( k = 2^m \) input streams and write to one output stream; hence the memory accesses are sequential unless \( m \) is too large
- SIMD friendly: most of the merge operations are done for integers; reading keys from records, which is costly with SIMD, only once per \( m \) stages

**Key-index approach Our approach \((m = 3)\)**
- encode \( key \) and \( index \) for each record into an integer,
- sort the key-index pairs with SIMD, and
- rearrange the records based on the sorted key-index pairs
- Efficient with SIMD due to gather for keys
- Cache friendly but NOT SIMD friendly

**Our approach**
- sort records directly without encoding into an integer
- Inefficient with SIMD due to gather for keys
- SIMD friendly but NOT cache friendly

**System**
- 2.9-GHz Xeon (SandyBridge) / RHEL 6.4 / gcc-4.8 / using SSE (128-bit SIMD)

**Sorting 512M records of 16 byte**

**Scalability with multiple cores**

**Sorting various numbers of 16-byte records**

**Sorting 16M records of various record sizes**

**Effect of number of ways \((k)\)**