The team... at 2am in the morning

Jamie Song - js4390@columbia.edu

Olesya Medvedeva - oam2113@columbia.edu

Ryan DeCosmo - rd2680@columbia.edu

Charis Lam - cl3257@columbia.edu
Concept: MapReduce

1. Large input data set. (ex. a book)

2. Data set gets split into chunks. (ex. small text files)

3. A function is applied to each chunk
   (ex. return the frequency of the word ‘hitchhiker’)

3. Aggregate all the results into one unit. (ex. 42)
Inspiration: Apache Hadoop
Expectations:

- BIIIIG DATA
- Multi-threaded on graphics card
- GPU-accelerated,
- In-memory
- Map-reduce replacement for single workstation users
reality...

miniMap:

Text processing language
Small-to-Medium Data
Sorta.. multi-threaded!
Lower overhead than the hadoop ecosystem

*Ideal? For projects / researchers
so how should it work?

miniMap()
works like MapReduce

miniMap(File* inputFile, void* splitter(), void* mapper(), File* context, void* reducer())
the pieces:

- File* inputFile: an input text file
- void* splitter(): function pointer to a function that splits the input file
- mapper(): function pointer to a user defined function
- File* context: an intermediate step that outsources RAM to disk
- reducer(): function pointer to a user defined function
Function headers

File** split_by_size(int x)
File** split_by_quant(int x)
File** split_by_regex(File*, String)

void mapper(File*, File*)

void reducer(File*)

void miniMap(input, splitter, mapper, context, reducer)
so how does it work?

Input File

Splitter Function
so how does it work?
so how does it work?
so how does it work?
so how does it work?
Architecture

Applied using threads
so how does it work?

Each file chunk has the map function applied to it
so how does it work?

Reducer combines data from mapper threads
File of clean, useful Data
Built-in Types

- ints
- bool
- float
- String
- void
- File
- Array
- Array pointer

```
A.Int -> i32_t
A.Bool -> i1_t
A.Float -> float_t
A.String -> str_t
A.Void -> void_t
A.File -> void_ptr
A.ArrayType(typ, size) -> (match typ with
  | A.Int -> array_t i32_t size
  | A.Float -> array_t float_t size
  | A.Bool -> array_t i1_t size
  | A.File -> array_t void_ptr size
  | _ -> raise (UnsupportedArrayType))

A.ArrayPointer(t) -> (match t with
  | A.Int -> pointer_t i32_t
  | A.Float -> pointer_t float_t
  | _ -> raise (IllegalPointerType))
```
Built-in functions.. links to C standard library!

Prints:
  print(), printb(), printbig(), **printstring()**

Splitters:
  **split_by_size()**, **split_by_quant()**, **split_by_regex()**

File:
  open(), readFile(), isFileEnd(), close()

String:
  strstr()
demo!
Our process:

- Weekly meetings
- Internal implementation goals
- Iterative cycle of concept and coding!
possible directions that Minimap could take:

- GPU acceleration using Nvidia CUDA
- Multi-Node Support (multiple multi-core PCs)
- Optimize File I/O - Sequential Offset (like Kafka)
THANKS FOR LISTENING
ANY QUESTION