**Taller 10**

MapReduce de Palabras

Crea un archivo llamado map\_reduce.cpp con el siguiente código:

|  |
| --- |
| #include <iostream>  #include <vector>  #include <map>  #include <algorithm>  #include <string>  #include <sstream>  // Función map: convierte un texto en pares de palabras y sus frecuencias  void map(const std::string &texto, std::map<std::string, int> &frecuencias) {  std::istringstream iss(texto);  std::string palabra;  while (iss >> palabra) {  ++frecuencias[palabra];  }  }  // Función reduce: combina las frecuencias de palabras de varios mapas  void reduce(const std::vector<std::map<std::string, int>> &mapas, std::map<std::string, int> &resultado) {  for (const auto &mapa : mapas) {  for (const auto &par : mapa) {  resultado[par.first] += par.second;  }  }  }  int main() {  // Datos de entrada: un vector de textos  std::vector<std::string> entradas = {  "MapReduce es un modelo de programación",  "MapReduce permite procesar grandes volúmenes de datos",  "MapReduce se popularizó con Hadoop",  };  // Aplica la función map a cada entrada  std::vector<std::map<std::string, int>> mapas(entradas.size());  for (size\_t i = 0; i < entradas.size(); ++i) {  map(entradas[i], mapas[i]);  }  // Aplica la función reduce a los mapas para obtener el resultado final  std::map<std::string, int> resultado;  reduce(mapas, resultado);  // Imprime el resultado  for (const auto &par : resultado) {  std::cout << par.first << ": " << par.second << std::endl;  }  return 0;  } |

Compila el programa utilizando g++:

|  |
| --- |
| g++ -o map\_reduce map\_reduce.cpp |

Ejecuta el programa:

|  |
| --- |
| ./map\_reduce |

El programa de ejemplo implementa un simple MapReduce para contar la frecuencia de palabras en un conjunto de textos. Puedes modificar las funciones map y reduce de acuerdo con tus necesidades y procesar datos más grandes y complejos según sea necesario.

Solicitud

Demostrar la mejora en rendimiento de utilizar mapReduce para contar la frecuencia de palabras en un conjunto de textos, respecto un conteo sin utilizar mapReduce, argumentando como esto podría servir para grandes volúmenes de datos, realizando un muestreo con diferentes escalas de volúmenes procesados, estimando como podría ser la mejora futura con X volumen de crecimiento.