Systemy równoległe i rozproszone - Klasyfikator dokumentów

Piotr Moszkowicz

21 czerwca 2021

Spis treści

1 Wstęp

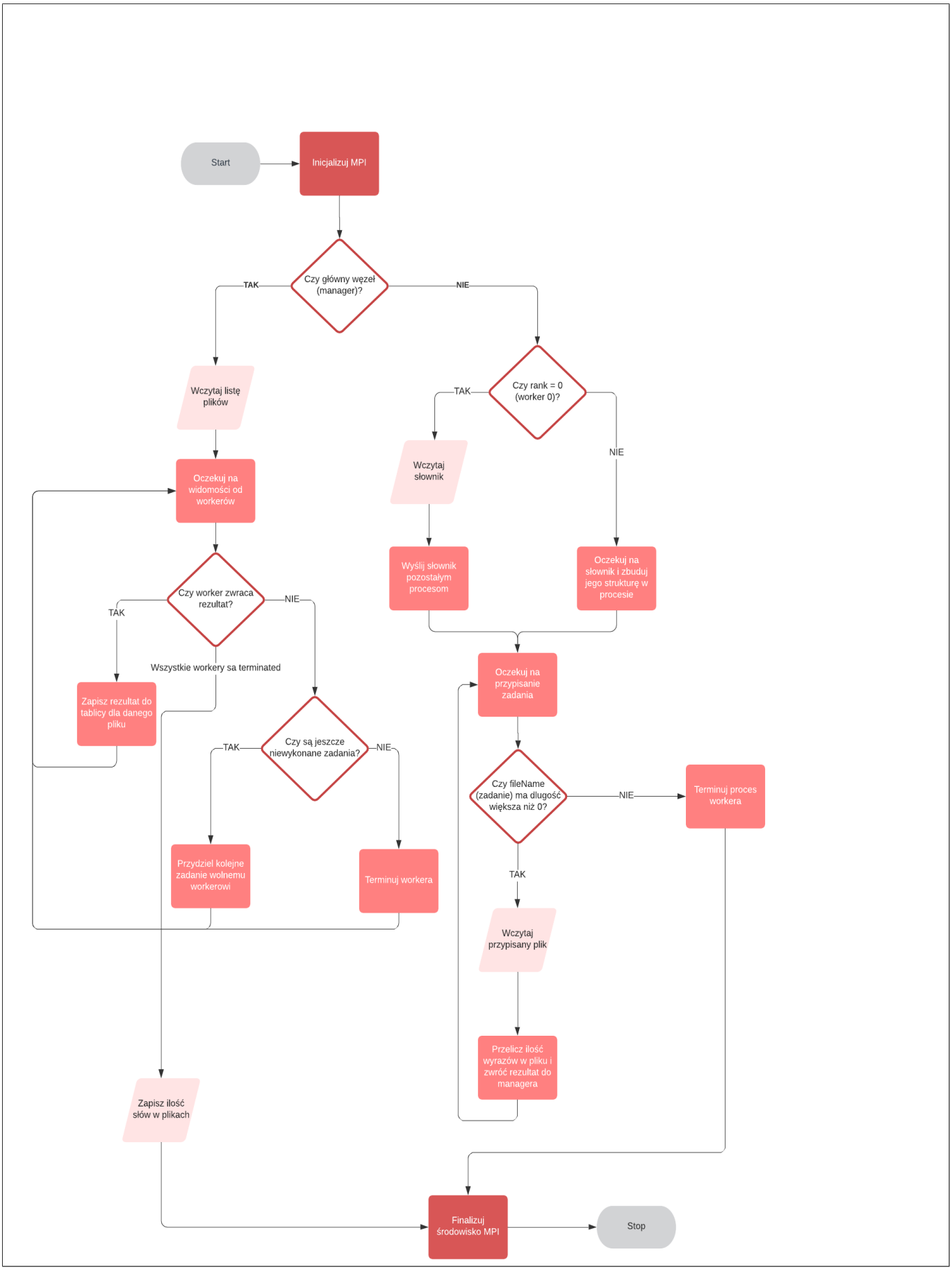
W ramach pierwszego projektu jego celem było zaprojektowanie współbieżnego programu z wykorzystaniem technologii MPI, który mógłby klasyfikować dokumenty.

1.1 Problem klasyfikacji

W tym wypadku klasyfikowane były pliki tekstowe formatów .html, .txt oraz .tex. Klasyfikacja odbywa się na podstawie pliku słownika *dict.txt*. W tym pliku znajdują się słowa kluczowe, które wyszukujemy we wszystkich pozostałych plikach w katalogu spełniających kryterium formatu. Następnie zliczana jest ilość słów w tych plikach, a finalnie rezultat jest sumowany.

2 Budowa programu

Program opiera się o mechanizm Manager - Workers. Jest to implementacja szeroko znanego wzorca map-reduce, wykorzystywanego między innymi w środowisku Spark. Procesowi zerowemu przypisujemy rolę managera zadań, którego zadaniem jest wczytanie listy dostępnych plików, a następnie rozdysponowanie ich pomiędzy workery. Zadaniem workerów jest policzenie słów w danym pliku (operacja "map"). Worker o ID 0 ma dodatkową rolę - na początku działania aplikacji jego zdaniem jest wczytanie słownika oraz przesłanie go do pozostałych workerów, które w ramach swojej dostępnej pamięci tworzą jego kopię w celu posiadania wiedzy jakie słowa kluczowe są poszukiwane. Sama implementacja wyszukiwania jest trywialna - zadany plik jest otwierany, a następnie słowo po słowie liczone są jego wyrazy. Finalnie z mapy wystąpień wyrazów wybieramy tylko te, które nas interesują. Gdy rezultaty z wszystkich procesów zostaną dostarczone managerowi ten wykonuje operację "reduce" - sumuje wystąpienia we wszystkich plikach i zapisuje wyniki do pliku wynikowego *result.txt*.



Rysunek 1: Schemat blokowy programu

Sam program został napisany w języki C++ w standardzie C++17 (głównie ze względu na nagłówek $filesystem$). Wersja środowiska MPI to 3.2.

3 Obsługa programu

Wraz z programem zostały dostarczone przykładowe pliki wejściowe oraz plik $makefile$, który pozwala na kompilację oraz uruchomienie programu. Procedura $makecompile$ kompiluje program, a z pomocą procedury $makerun$ możemy uruchomić program (domyślnie na 8 węzłach). Przed tym należy jednak skonfigurować środowisko MPI wykorzystując skrypty dostępne na serwerze Taurus.

Dla przykładowych danych testowych (umieszczonych w archiwum) osiągnięto następujący rezultat: $7moszkowicz@stud206 - 001 : \ /WFiIS - SRiR - 2021/proj1\$catresult.txt112$