Zaawansowane Programowanie w C++

Zadanie dodatkowe

Temat projektu: biblioteka Ranges (views)

Mateusz Krakowski

15 maja 2023

1 Oryginalny Temat Projektu

Opisanie biblioteki ranges dodanej w C++20 i poszerzonej w C++23. Praca będzie skupiać się na opisaniu przydatnych funkcji z biblioteki ranges służących do pracy z kolekcjami. Na wstępie wytłumaczę różnicę między ranges::range i ranges::view.

2 Różnica między ranges::range i ranges::view

Range to zakres definiowany przez obiekt iteratora (służący do iterowania po elementach) oraz wskaźnik na ostatni element (ang. sentinel). Każdy iterator, szereg, pojemnik można nazwać zakresem, jeśli posiada on metody begin() oraz end(). View natomiast pozwala nam na tworzenie strumieniowych widoków na powyższych zakresach. Widok nie tworzy nowego zakresu, jedynie kożysta z danych zapisanych w zakresie i pozwala na funkcyjną manipulację nimi. Dużym plusem widoków jest to, że wszystkie transformacje są wykonywanie leniwie (ang. lazy evaluation), co oznacza, że przemiany na danym elemencie widoku są wykonywane dopiero wtedy, gdy żądamy dostępu do ów elementu. Obrazuje to przykład zawarty w funkcji rangesViewsDiffShowcase(). Przy podanym dużym wektorze danych widok okazał się szybszy od tworzenia nowego wektora o przynajmniej 100 razy.

3 Przydatne funkcje widoków

Starałem się wybrać funkcje najbardziej przydatniejsze pod względem analizy danych i trenowania modeli sztucznej inteligencji. Większość z poniższych funkcji są dostępne domyślnie w popularnych językach programowania wspierających analizę i przetwarzanie danych takich jak Python czy R.

3.1 Filter

Pozwala nam na stworzenie widoku na zakresie i przeprowadzenie operacji filtrowania pod względem warunku logicznego. Innymi słowy, pozwala nam dostać się do elementów w zakresie o danych cechach logicznych. std::views::filter może okazać się bardzo przydatne jeśli chcemy otrzymać widok na zakres elementów, dla których atrybut mieści się w danym zakresie, np. chcemy otrzymać widok na zakres osób w przedziale wiekowym 18-26 lat.

3.2 Transform

Pozwala nam na stworzenie widoku na zakresie i jednoczesnym przeprowadzeniu transformacji np. funkcji podnoszącej wszystkie wartości do potęgi drugiej. Transform może również się przydać gdy chcemy zastosować transformacje danych, np. zastosowanie funkcji aktywacji dla wszystkich elementów zakresu.

3.3 Take i Drop

Take i Drop pozwala nam na proste stworzenie widoku na zakres wybierając lub odrzucając zadaną ilość elementów. Wraz z pomocniczą funkcją calculateSplitPoint() możemy podzielić zakres na dwa podzakresy w zadanych proporcjach.

3.4 Reverse

Reverse pozwala nam w łatwy sposób otrzymać widok na zakres elementów w odwróconej kolejności, możemy np. dostać się do ostatnich trzech wartości zakresu najpierw przeprowadzając na nim reverse, a następnie take(3).

3.5 Zip

Z uwagi na to że std::views::zip pojawił się dopiero w gcc13 wydzieliłem go do oddzielnego pliku. Funkcja zip pozwala nam iterować po elementach zakresu w tej samej funkcji for(). Jest ona przydatna np. gdy chcemy ocenić sprawność naszego modelu, jako przykład podałem metodę obliczającą macierz pomyłek z wykorzystaniem iteracji po elementach dwóch zakresów.

4 Źródła wiedzy

- https://ericniebler.github.io/range-v3/
- https://en.cppreference.com/w/cpp/ranges
- https://github.com/ericniebler/range-v3