В математике, метрика - это функция, которая измеряет расстояние между элементами некоторого пространства. Она должна удовлетворять трем основным свойствам: неотрицательность, симметричность и неравенство треугольника. В случае алгоритма Вагнера-Фишера, мы используем метрику для измерения расстояния между двумя строками - количество операций (вставок, удалений и замен), которые необходимо выполнить, чтобы превратить одну строку в другую.

Алгоритм Вагнера-Фишера основывается на динамическом программировании и позволяет вычислить расстояние между двумя строками. Он работает следующим образом:

Инициализируем матрицу размером (M+1) x (N+1), где M и N - длины строк.

Заполняем первую строку числами от 0 до N (количество операций вставки).

Заполняем первый столбец числами от 0 до M (количество операций удаления).

Для каждой позиции (i, j) внутри матрицы, вычисляем минимальную стоимость операции (удаление, вставка или замена) и записываем ее в текущую ячейку.

В результате получаем значение в правом нижнем углу матрицы, которое представляет собой минимальное расстояние между строками.

Вместо двумерного массива в тексте класса WF используется одномерный массив pm, который представляет собой линеаризованную версию матрицы размером (M+1) x (N+1). Это позволяет упростить работу с памятью и избежать проблем повторной инициализации, копирования и присваивания. Массив pm выделяется и освобождается в конструкторе и деструкторе класса соответственно, а доступ к элементам матрицы осуществляется через методы Get и Set.

Особенности метода Distance:

- Использует алгоритм Вагнера-Фишера для вычисления минимального расстояния между двумя строками.

- Инициализирует матрицу pm и заполняет ее значениями, соответствующими операциям вставки, удаления и замены.

- Вычисляет минимальную стоимость операций для каждой позиции матрицы, записывая ее в ячейки.

- Возвращает значение в правом нижнем углу матрицы, которое представляет собой минимальное расстояние между строками.

Чтобы улучшить метод Distance в случае, когда ожидается, что M и N могут отличаться в несколько раз, можно использовать оптимизацию под названием "диагональный просмотр" (diagonal scan). Вместо заполнения всей матрицы pm, алгоритм будет вычислять только необходимые диагональные элементы, начиная от верхнего левого угла и двигаясь по диагонали к правому нижнему углу. Это позволяет сократить количество операций и использование памяти, особенно при больших значениях M и N.

Использование диагонального просмотра позволяет значительно улучшить производительность алгоритма, особенно когда M и N имеют существенное отличие в размере.