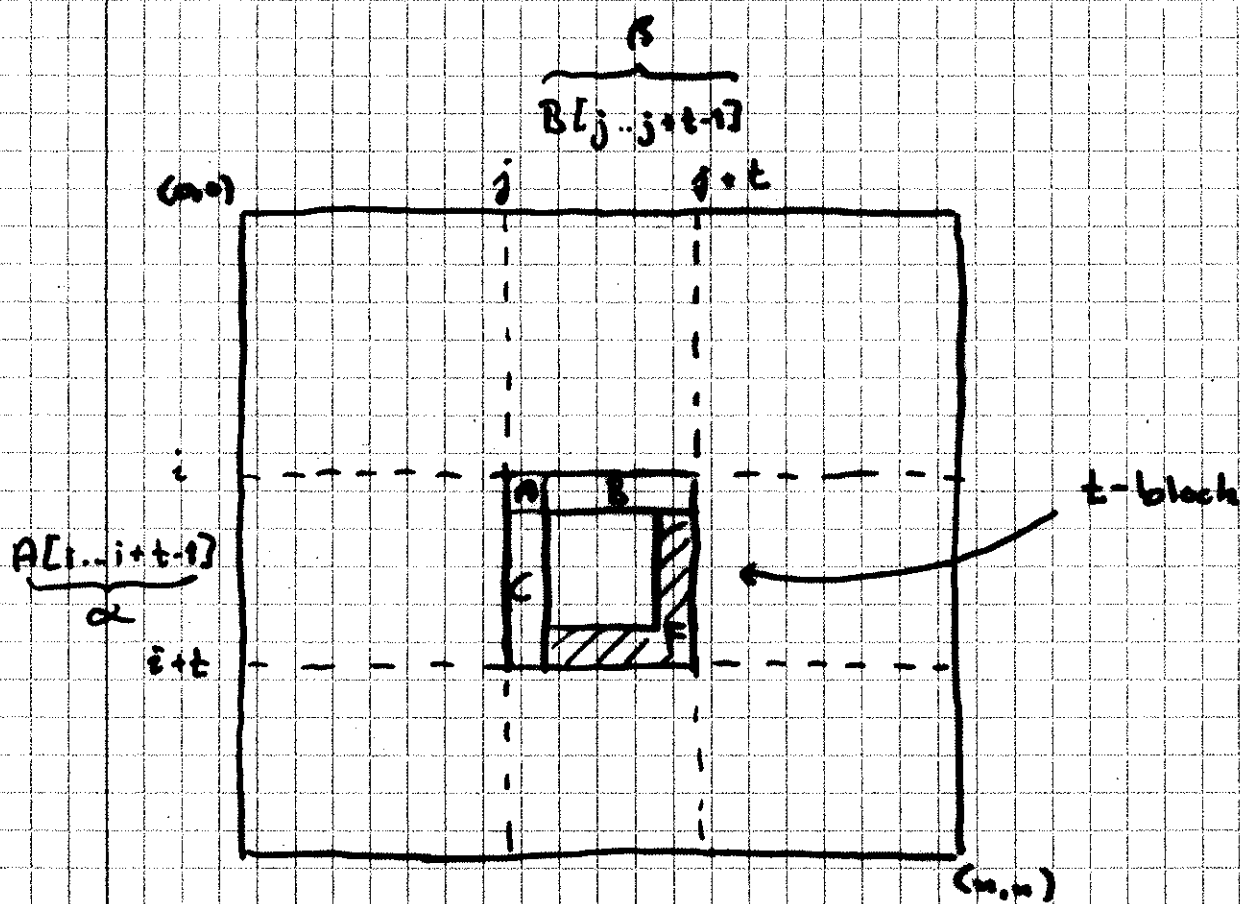


The D-table

Unit-cost edit-distance:

$$D(i, j) = \min \begin{cases} D(i-1, j-1) + \delta(A_i, B_j) & i > 0, j > 0 \\ D(i-1, j) + 1 & i > 0, j = 0 \\ D(i, j-1) + 1 & i = 0, j > 0 \\ 0 & i = 0, j = 0 \end{cases}$$



Block-function:

$$b(\alpha, \beta, A, B, C) = F$$

size of input and output is $O(t)$

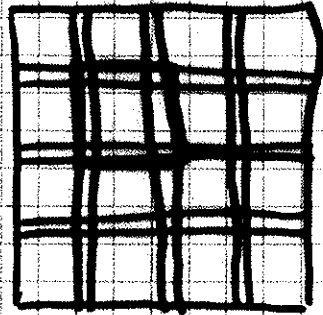
Algorithm idea

1) Cover the D-table with t-blocks such that they overlap one row/column.

2) Initialize first row/column.

3) Fill the table row-by-row using the block-function

4) return $D(n, m)$



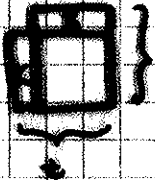
If we assume that the block function has been precomputed, then output F can be fetched in time $O(t) \dots$

Running time:

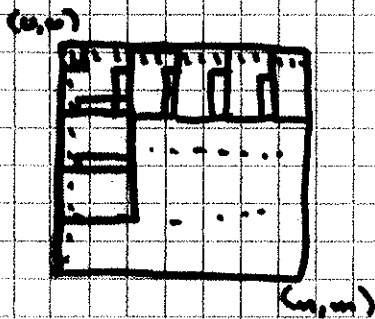
$\Rightarrow \left(\frac{n^2}{t^2} \right)$ blocks taking time $O(t)$ each, i.e.

$\Rightarrow \left(\frac{n^2}{t} \right)$ + "time to precompute block-function"

The four-Russian algorithm

1) Cover the D-block with  blocks.

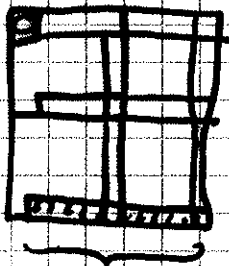
such that the first/last row and column overlap



2) Initialize first row/column with offsets $1, \dots, 1$

3) Use precomputed block-function to compute blocks in a row-wise manner...

4) The last row contain the offsets from $D(n, 0)$



$$\sum \text{"offsets"} = Q$$

$$D(n, n) = D(n, 0) + Q = \underline{\underline{n + Q}}$$