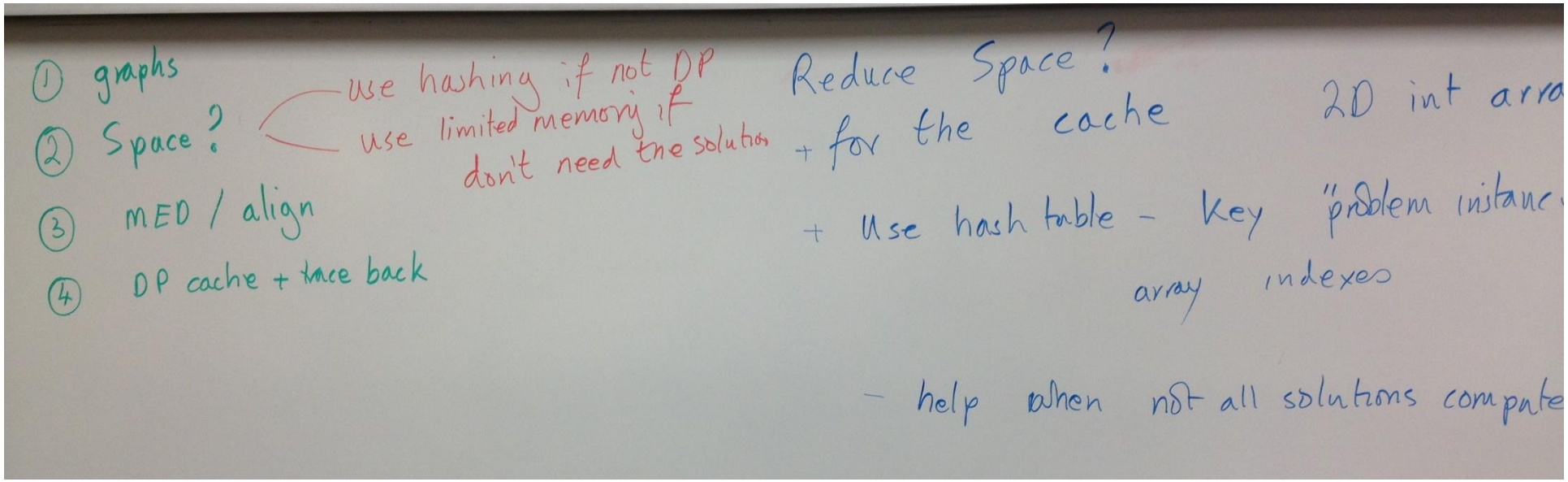
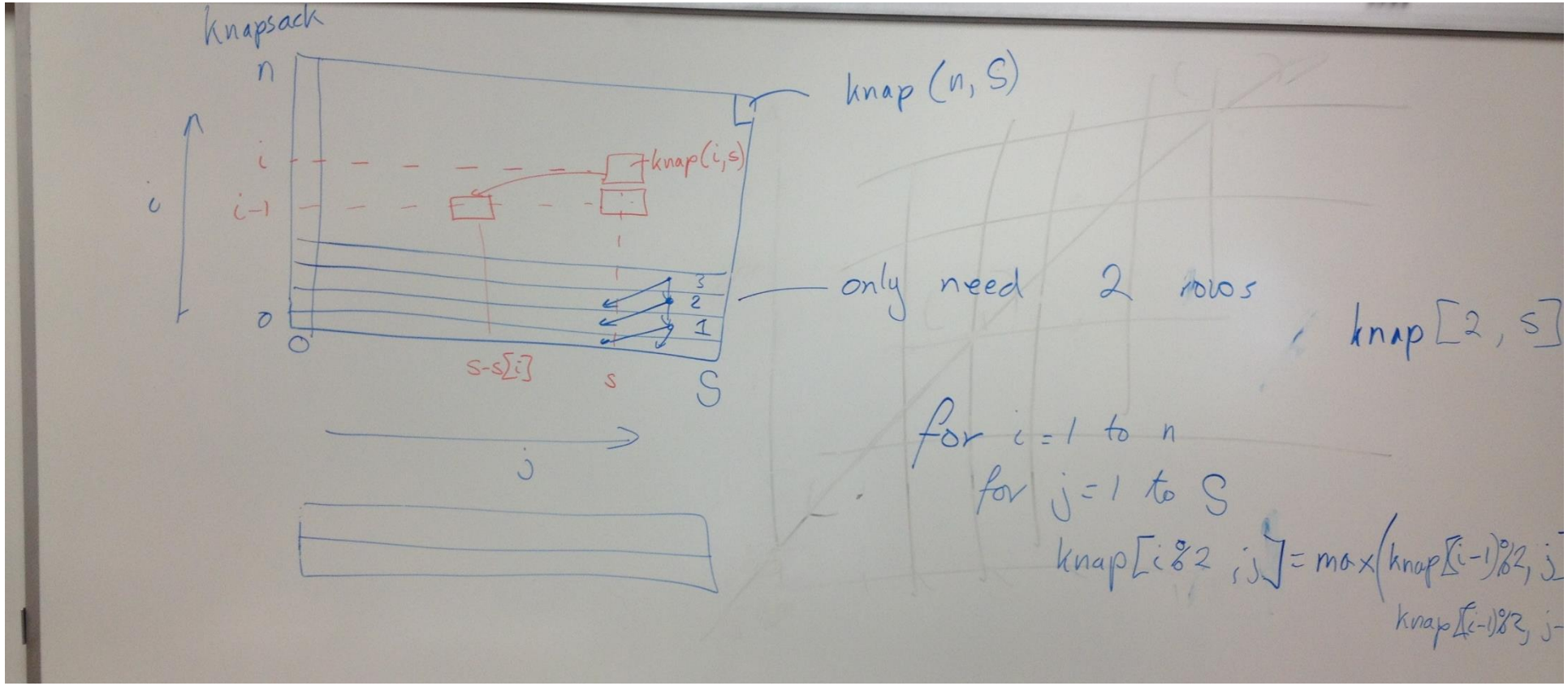
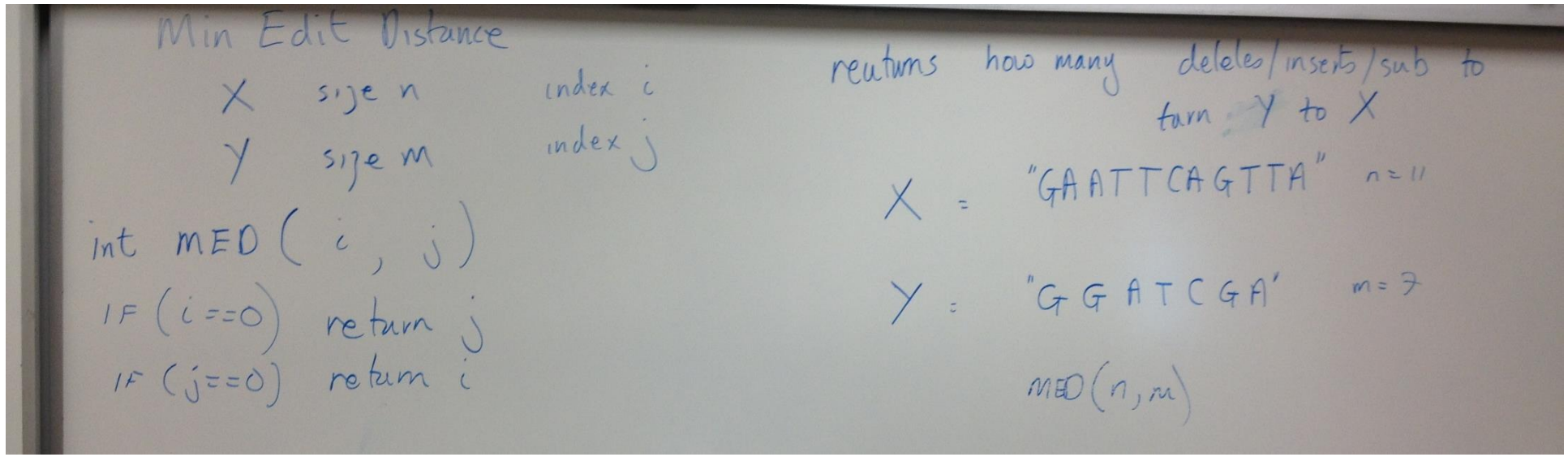
Cs5050 notes 01 23 14



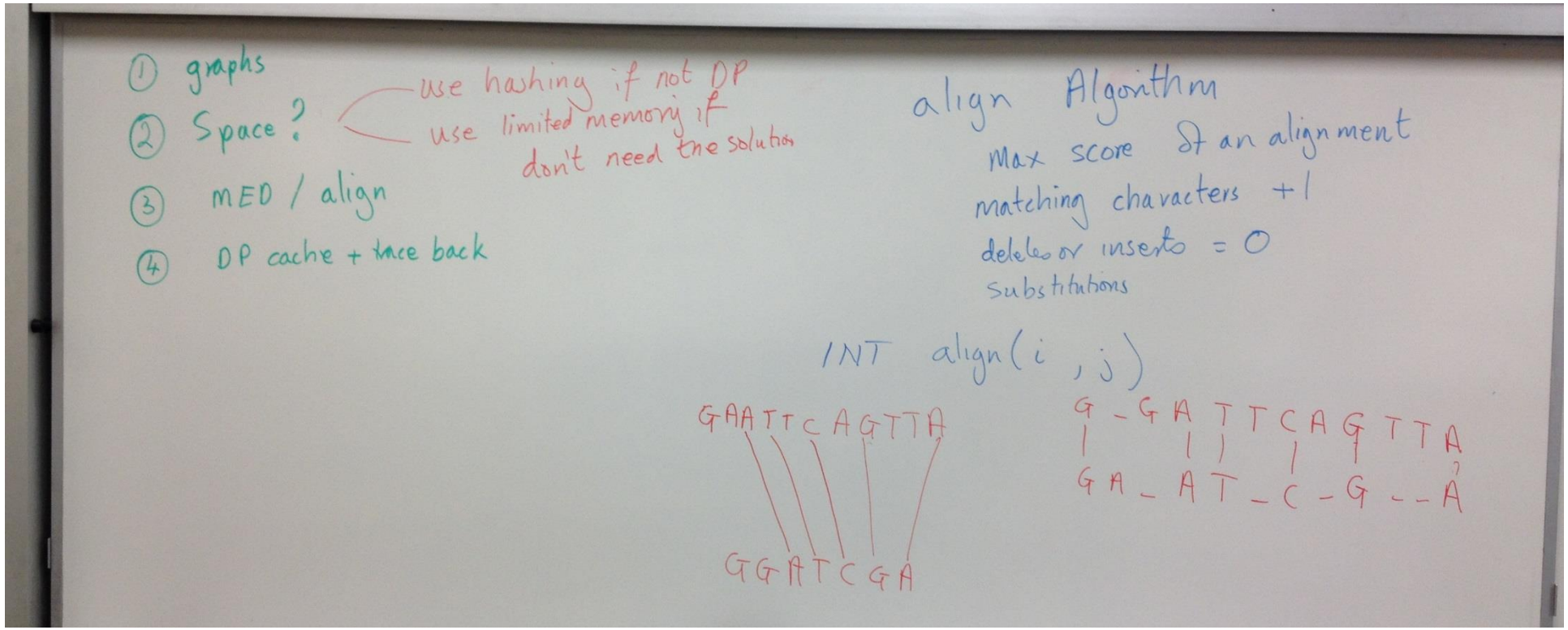
DP and Caching algorithms are often “space bound”
There space requirements grows too fast as a function of the problem size
In knapsack n objects S size, space needs are proportional to n * S
Solutions:
Use a hash table with caching if not all solutions needed



In DP knapsack the computation of the current row only needs the solutions contained in the previous row
Only two rows are needed to compute the value of the best solution
Easy code change. Allocate the array with two rows indexed 0 and 1
Change indexing to i%2 and (i-1)%2

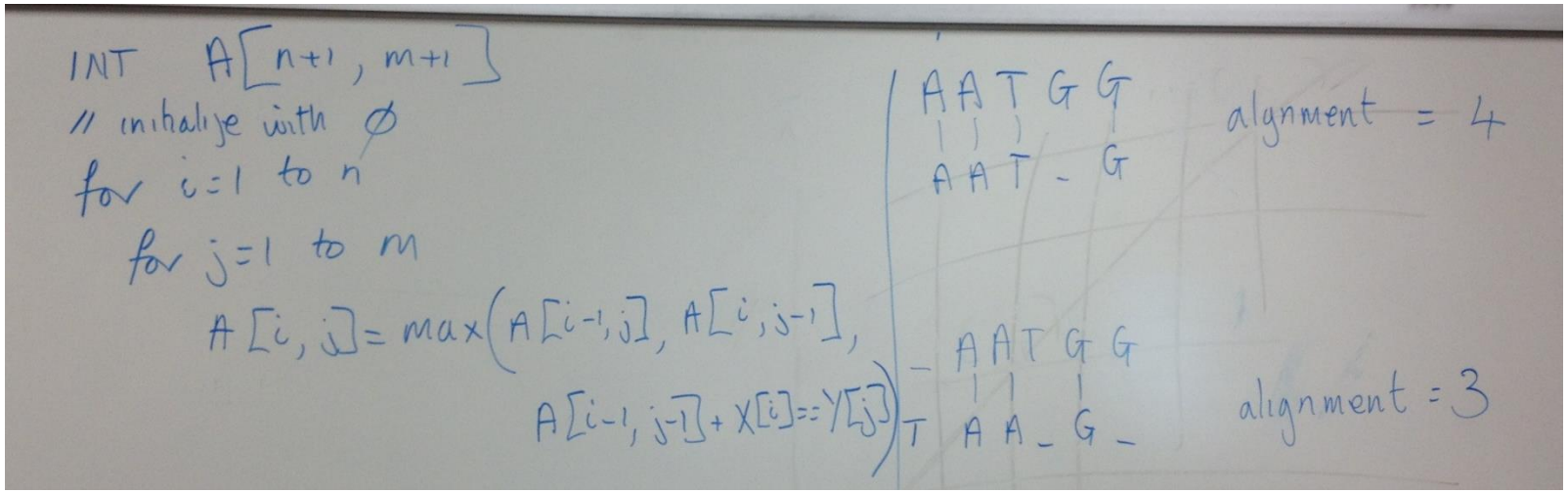


New problem: minimum edit distance

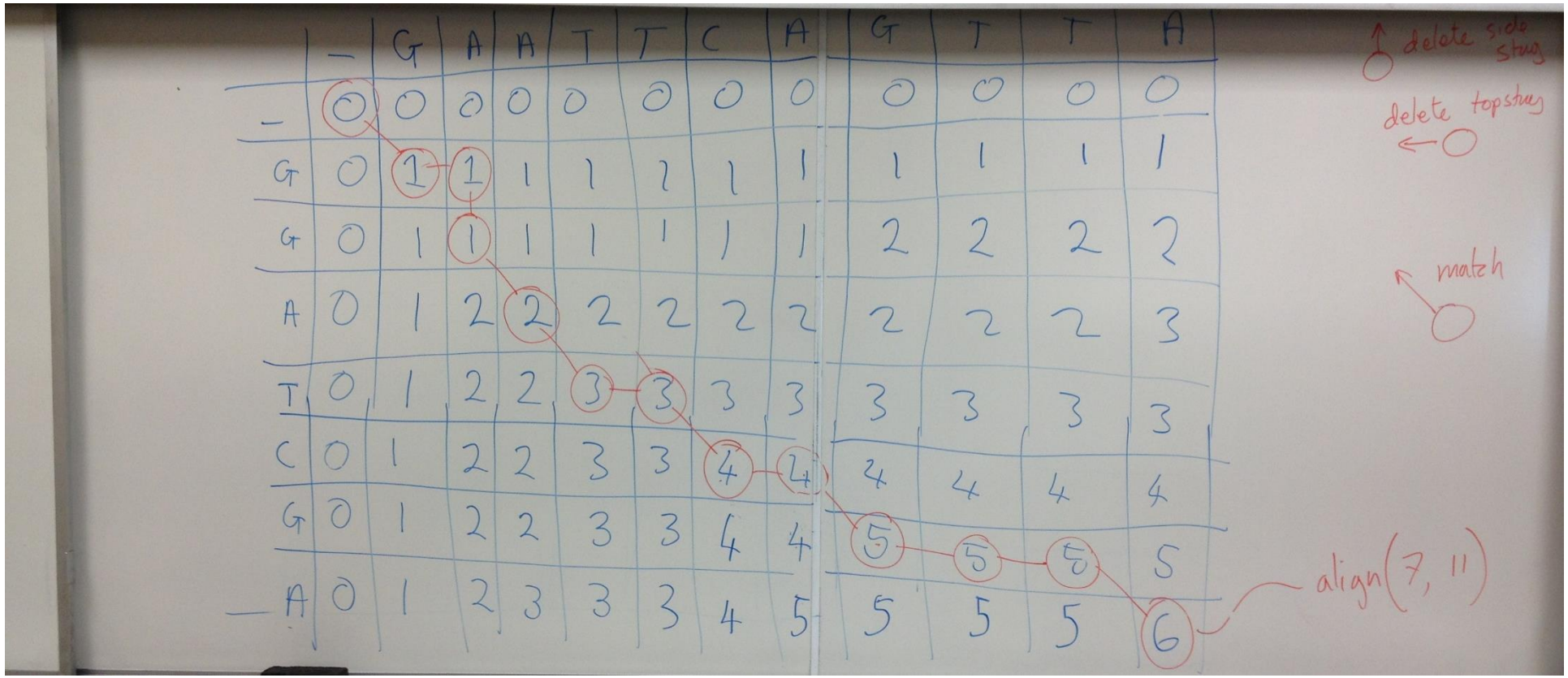


String alignment definition and solution representations (in red)
The solution on the left shows from left to right

G-G aligned
A aligned with a gap (delete A from top string)
gap aligned with G (delete G from bottom string)



Simple dynamic program developed from the recursive solution



Example trace back through the cache array