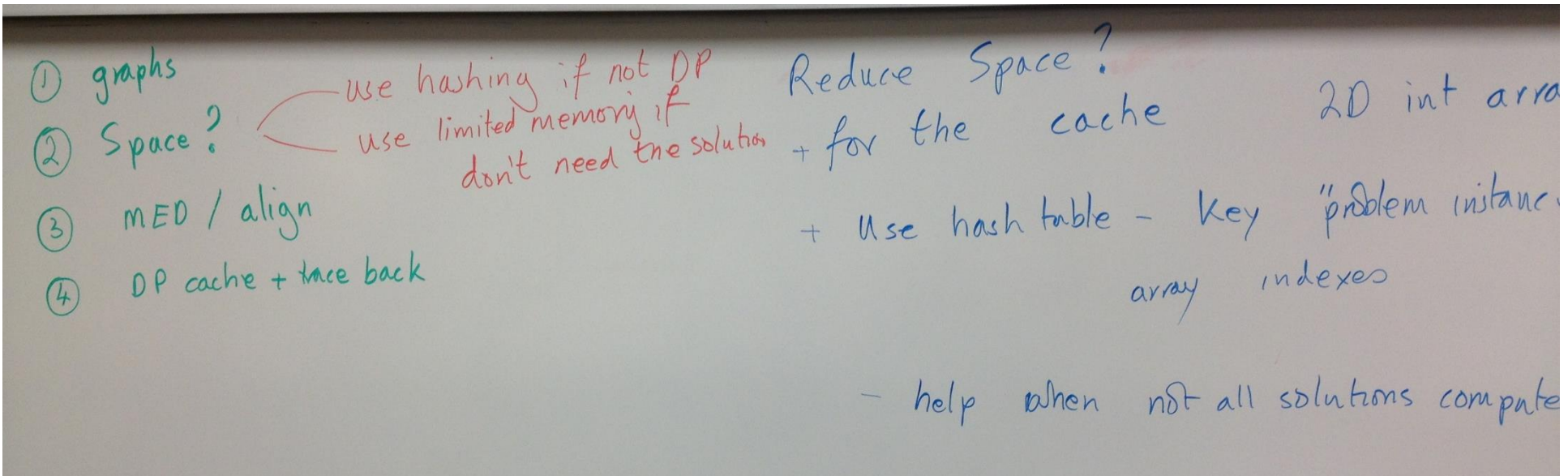


Cs5050 notes 01 23 14



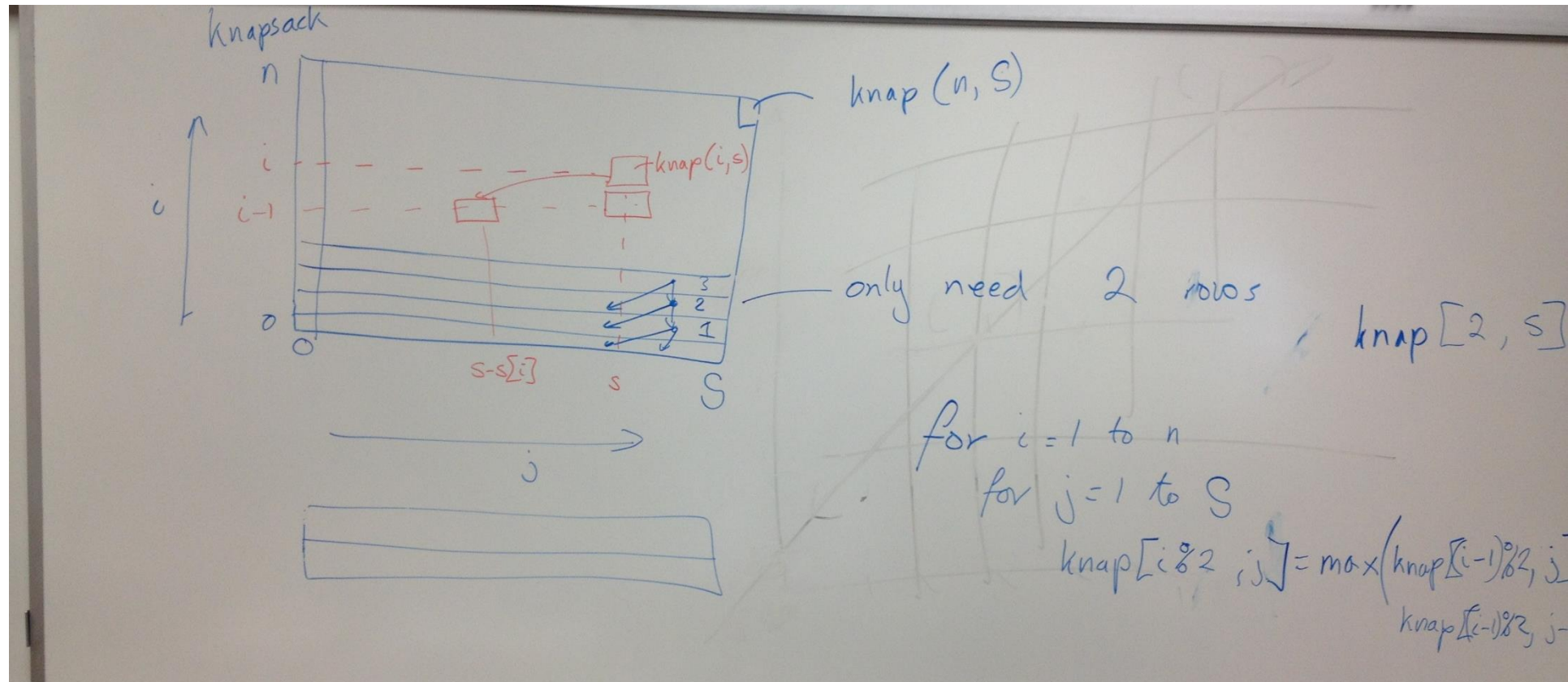
DP and Caching algorithms are often "space bound"

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 $(i-1)\%2$

Min Edit Distance

X size n index i
Y size m index j

```
int MED ( i, j )  
if ( i == 0 ) return j  
if ( j == 0 ) return i
```

returns how many deletes/inserts/sub to
turn Y to X

X = "GAATTCAGTTA" n = 11

Y = "GGATCGA" m = 7

MED(n, m)

New problem: minimum edit distance

- ① graphs
- ② Space? { use hashing if not DP
use limited memory if
don't need the solution
- ③ MED / align
- ④ DP cache + trace back

align Algorithm
 Max score of an alignment
 matching characters +1
 delete or insert = 0
 substitutions

INT align(i, j)

GAATTCAGTTA
 | | | | |
 GGATCGA

G - G A T T C A G T T A
 | | | | |
 G A - A T - C - G - - A

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)

INT A[n+1, m+1]

// initialize with \emptyset

for i=1 to n

for j=1 to m

A[i, j] = max(A[i-1, j], A[i, j-1],

A[i-1, j-1] + X[i] == Y[j])

A A T G G
| | |
A A T - G

alignment = 4

A A T G G
| | |
A A - G -

alignment = 3

Simple dynamic program developed from the recursive solution

	-	G	A	A	T	T	C	A		G	T	T	A
-	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1	1
A	0	1	1	1	1	1	1	1	2	2	2	2	2
T	0	1	2	2	3	3	3	3	3	3	3	3	3
C	0	1	2	2	3	3	4	4	4	4	4	4	4
G	0	1	2	2	3	3	4	4	5	5	5	5	5
A	0	1	2	3	3	3	4	5	5	5	5	6	6

delete side string
 delete top string
 match
 align(7, 11)

Example trace back through the cache array