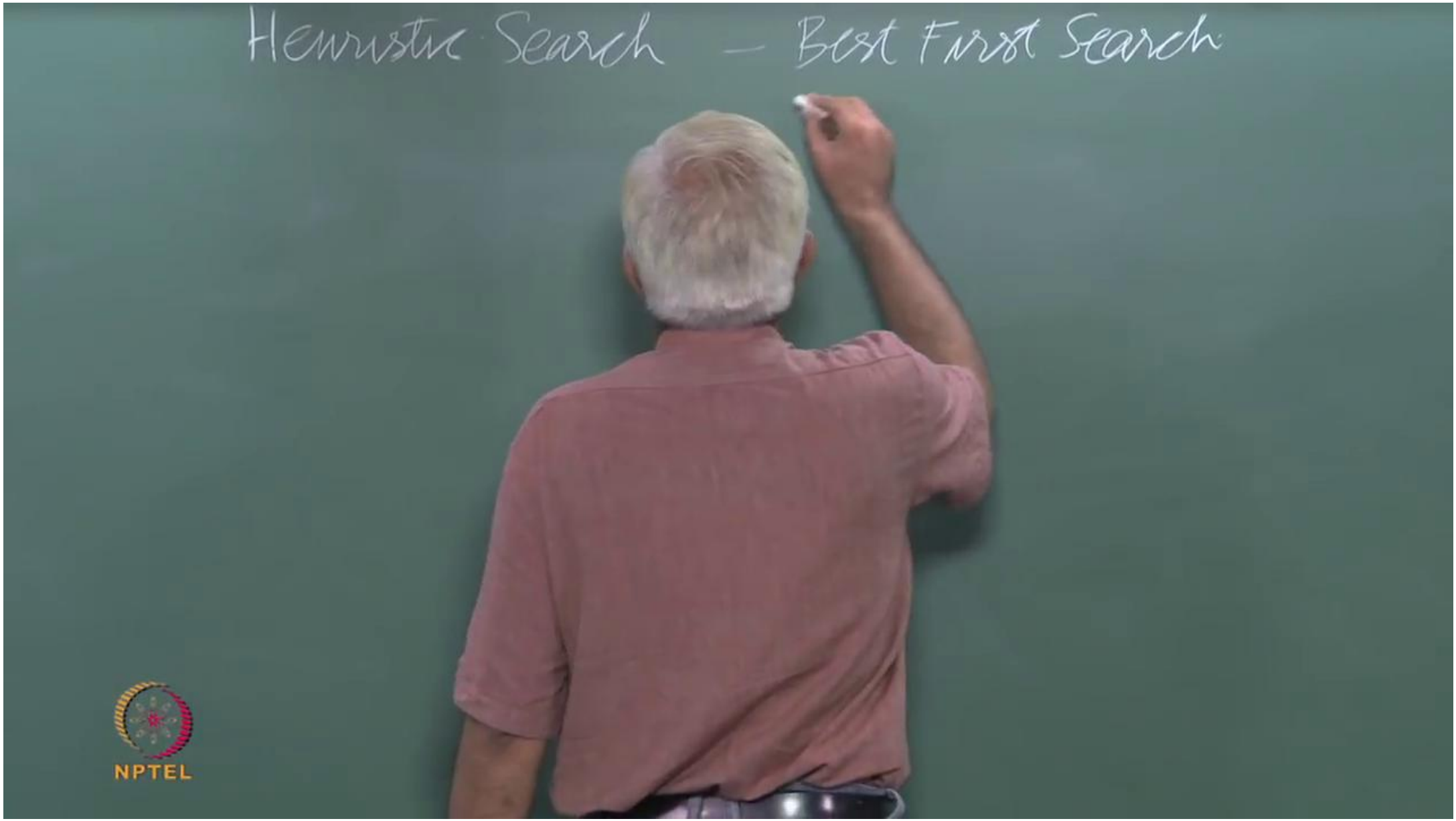
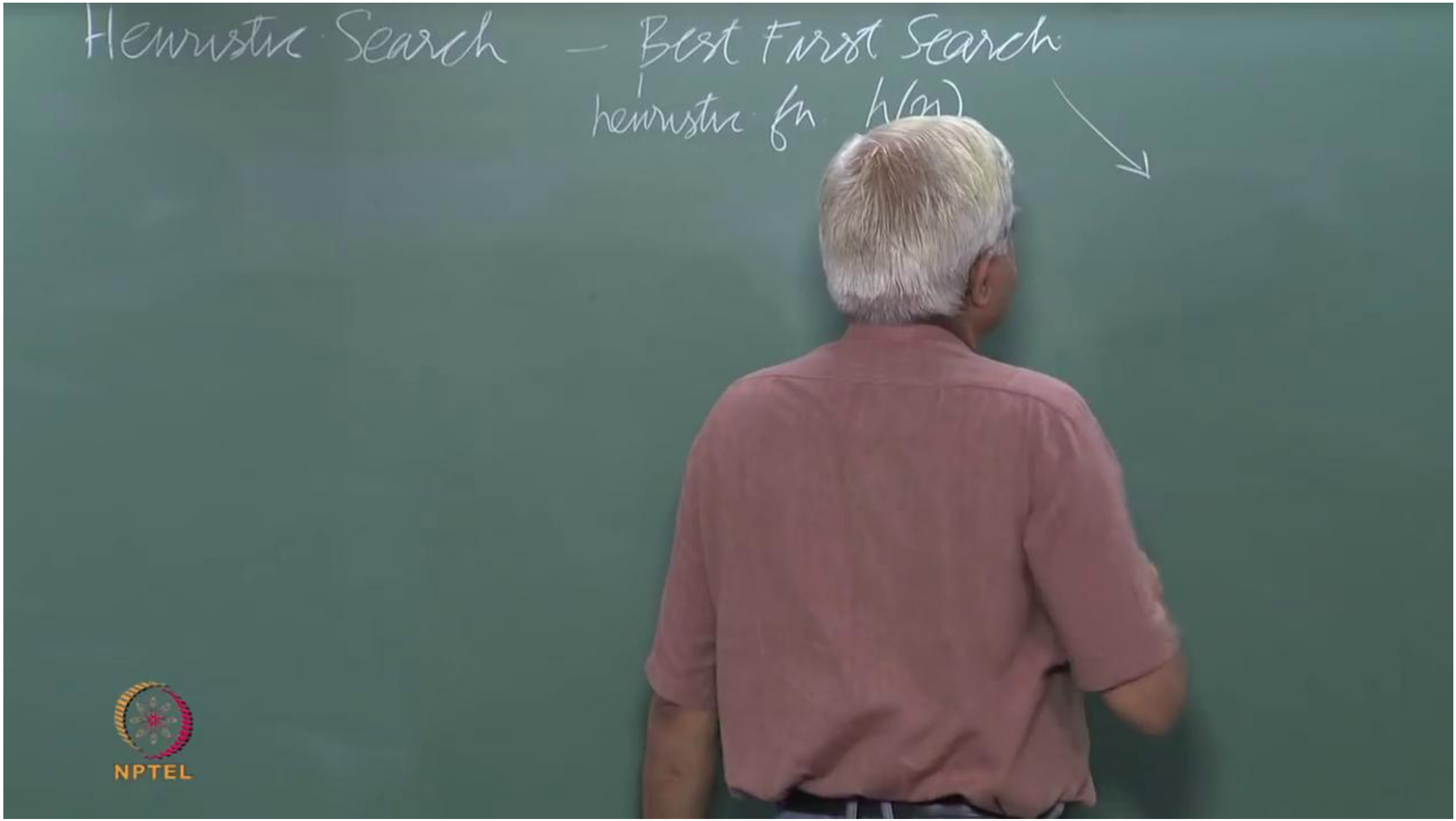
Video Lectures On

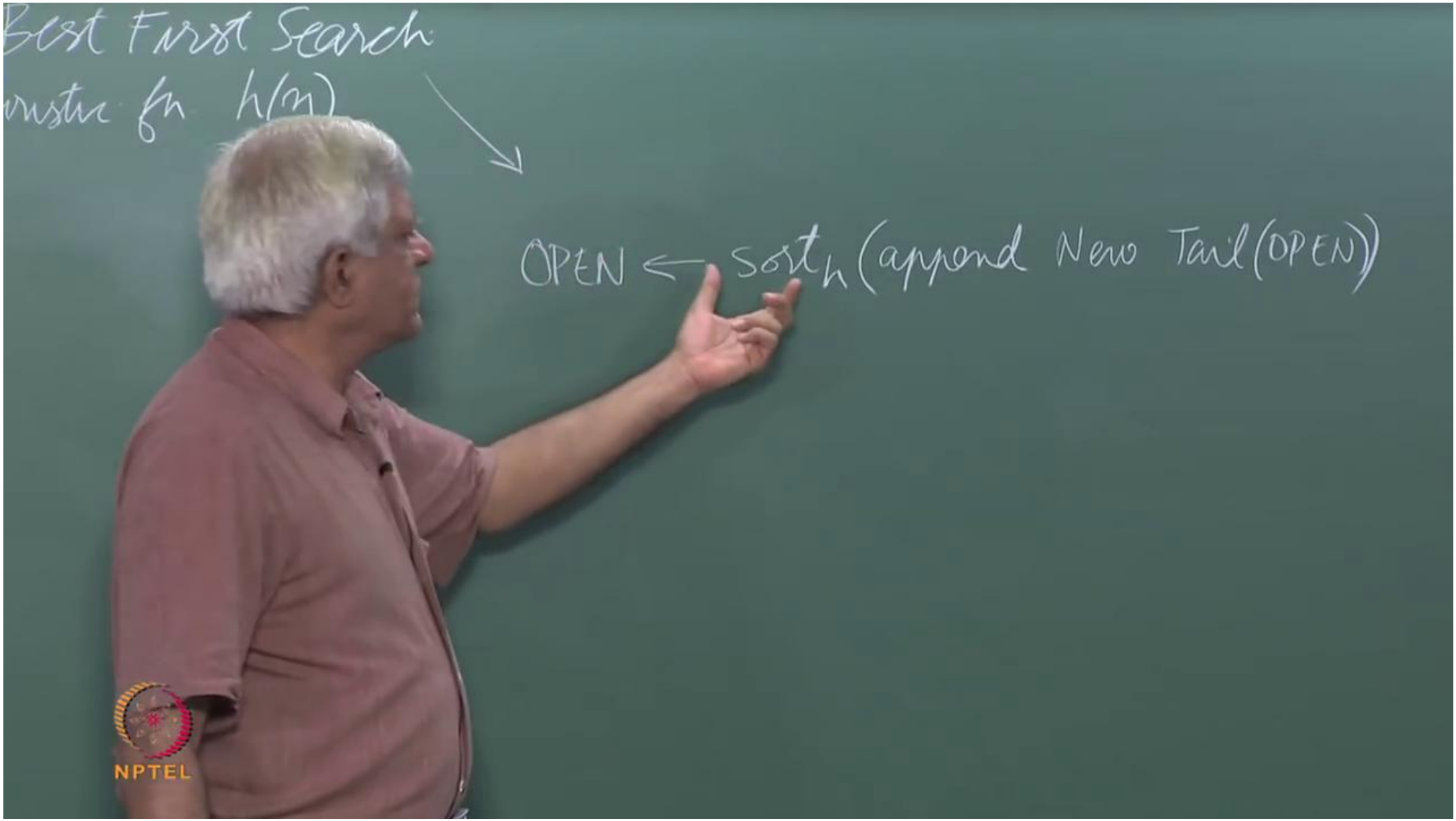
Artificial Intelligence

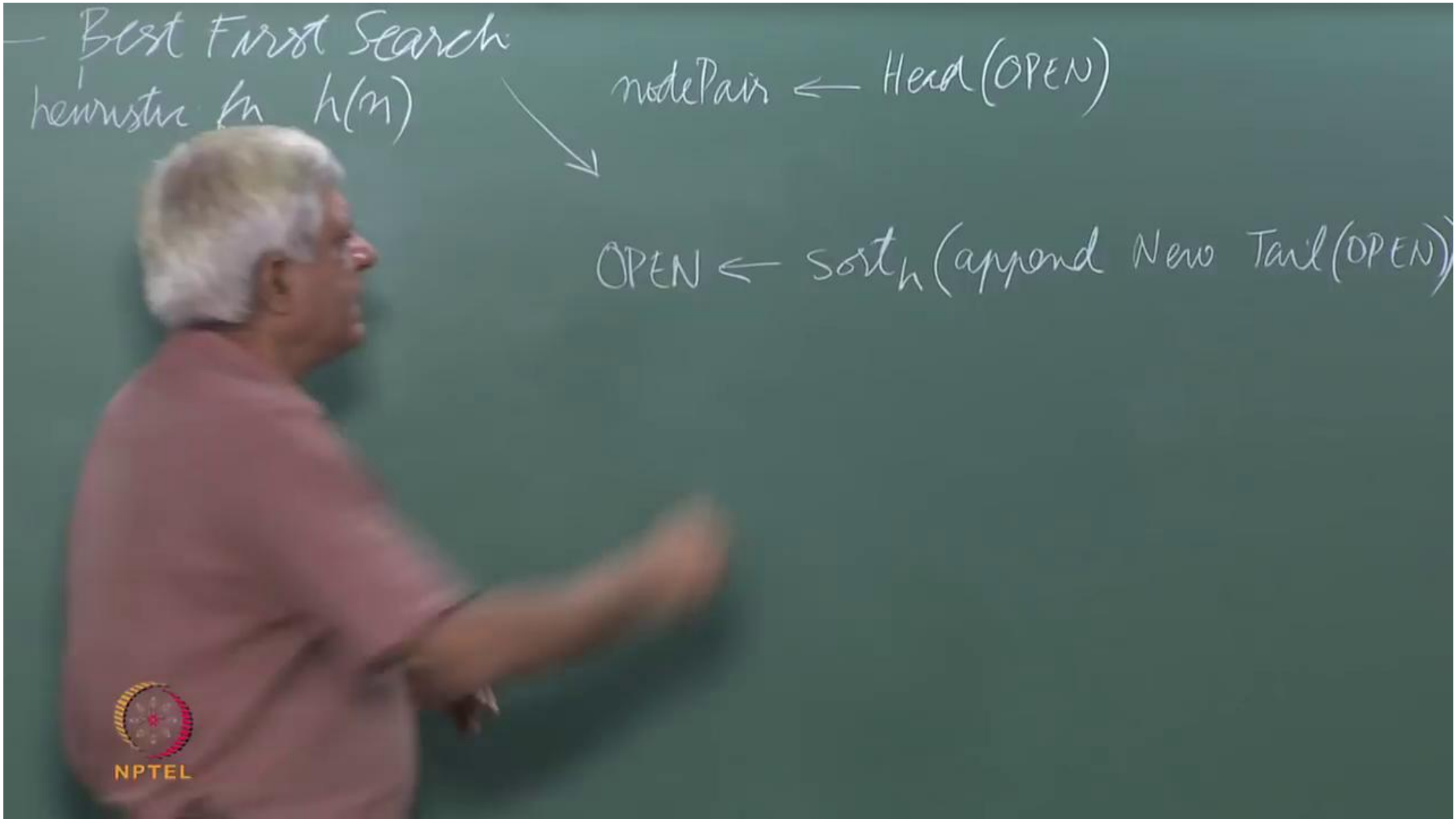
Lecture 10
Hill Climbing

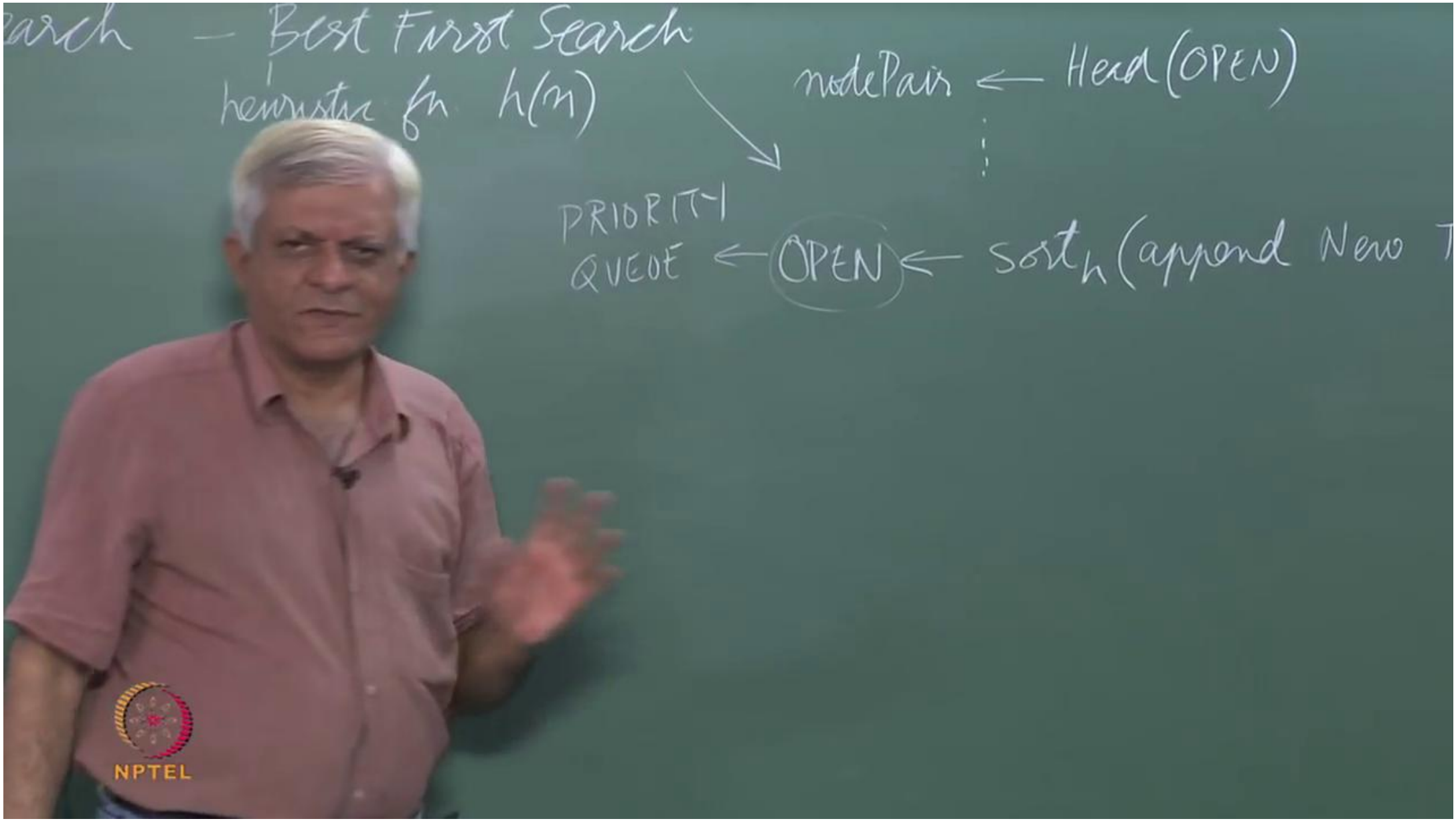
Prof. Deepak Khemani
Department of Computer Science and Engineering
IIT Madras

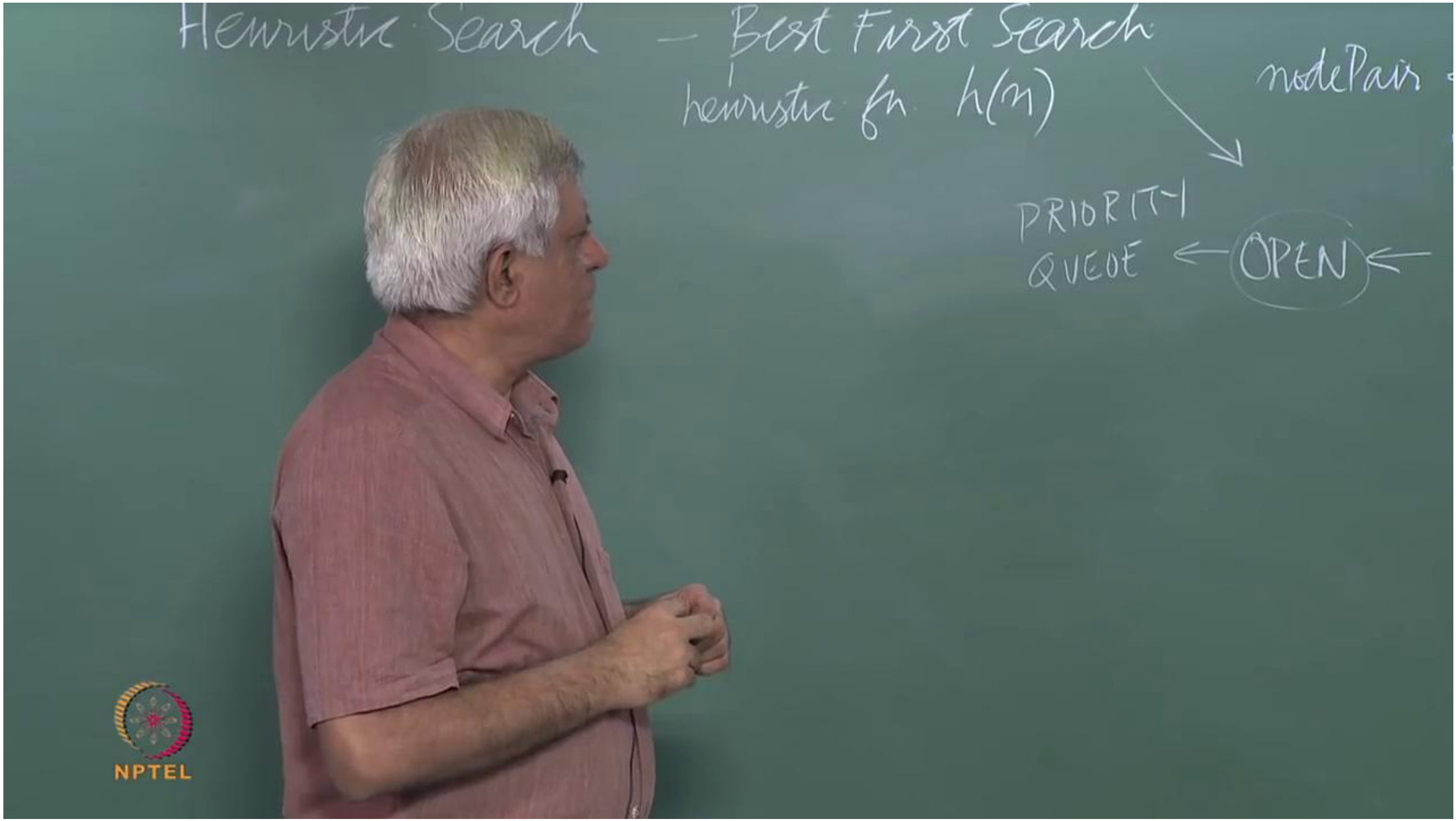

Heuristic Search – Best First Search
NPTEL

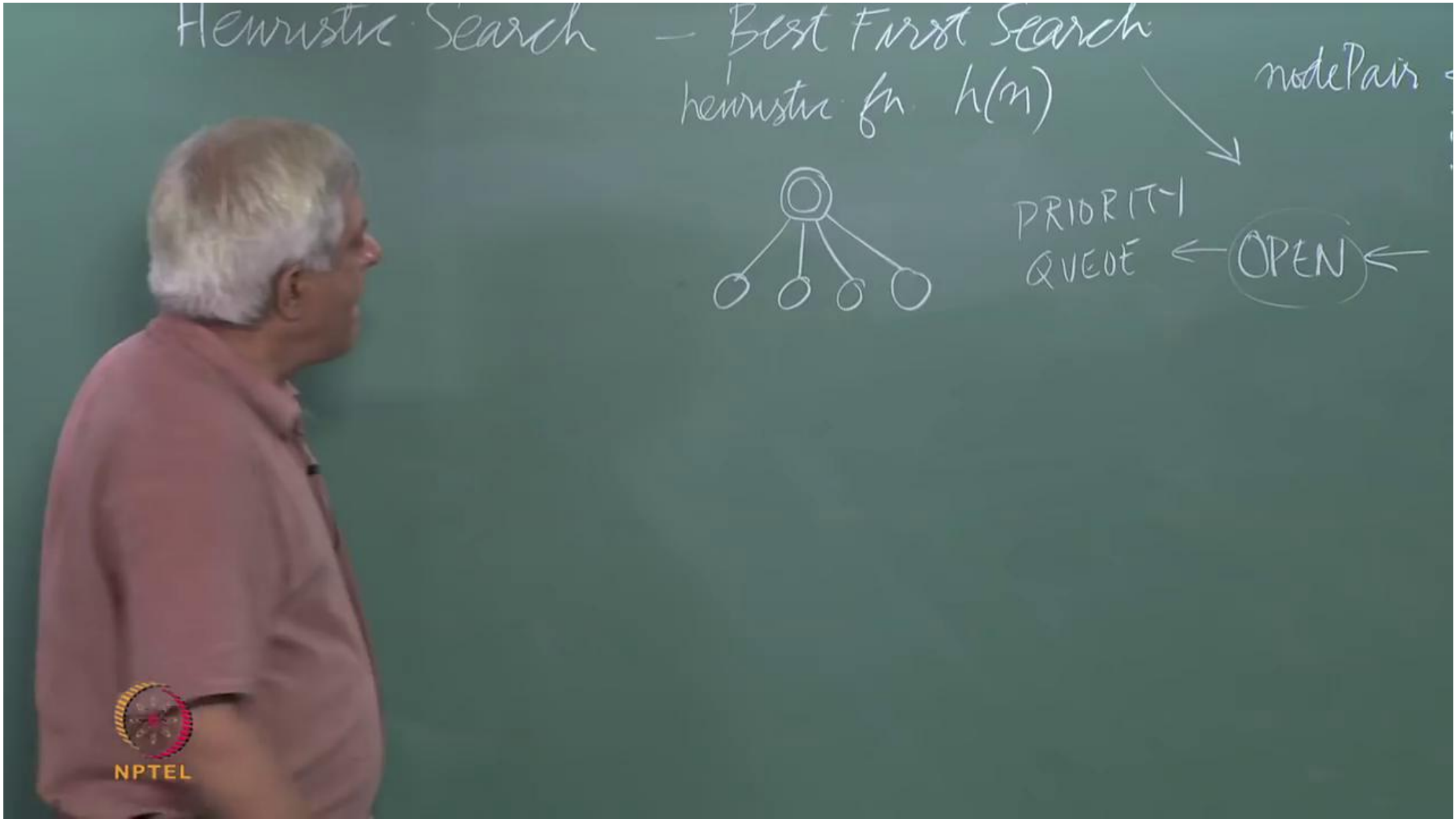

Heuristic Search – Best First Search
heuristic fn h(n)
NPTEL

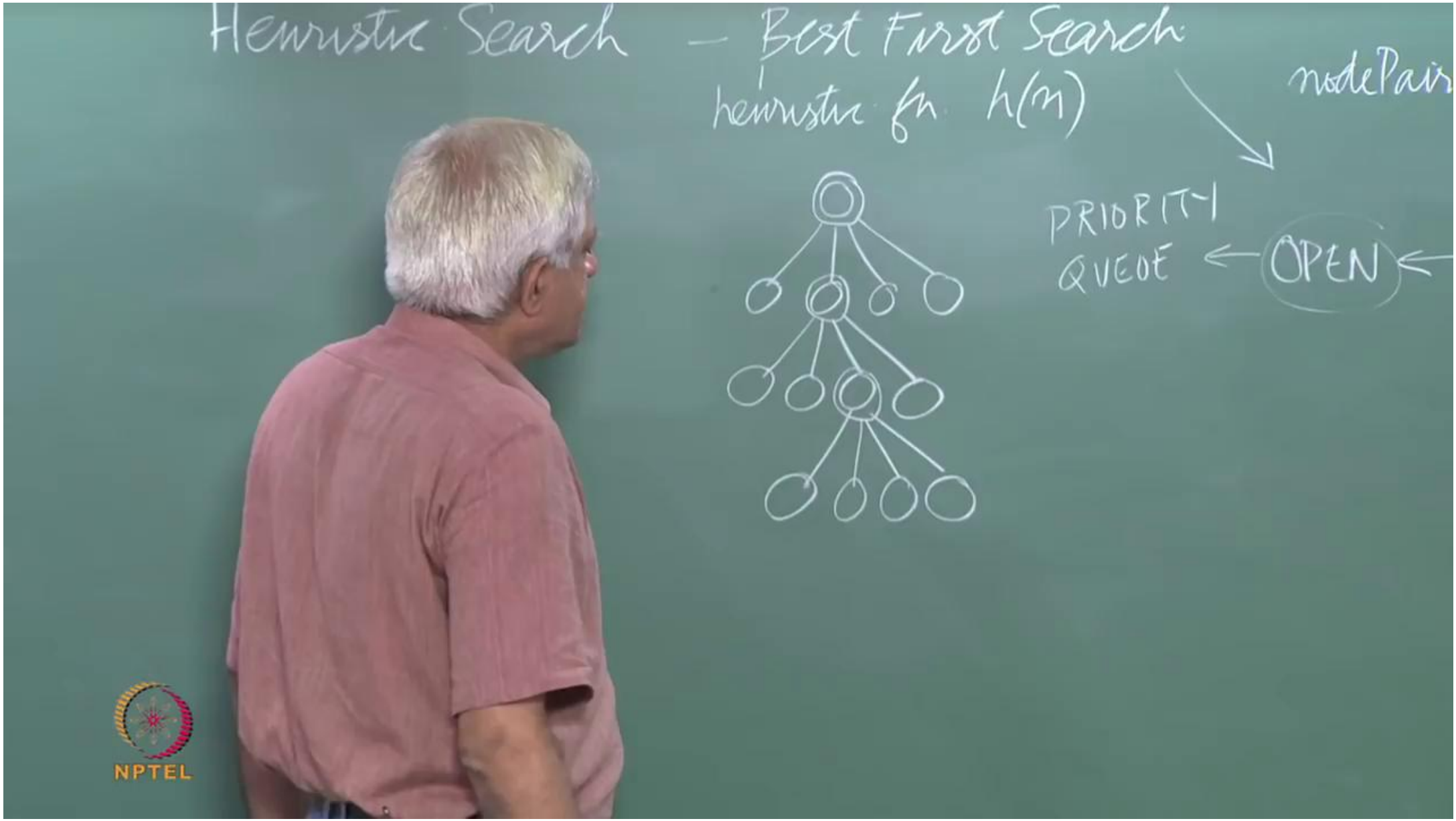

Best First Search
OPEN ← sort_h (append New Tail(OPEN))
NPTEL

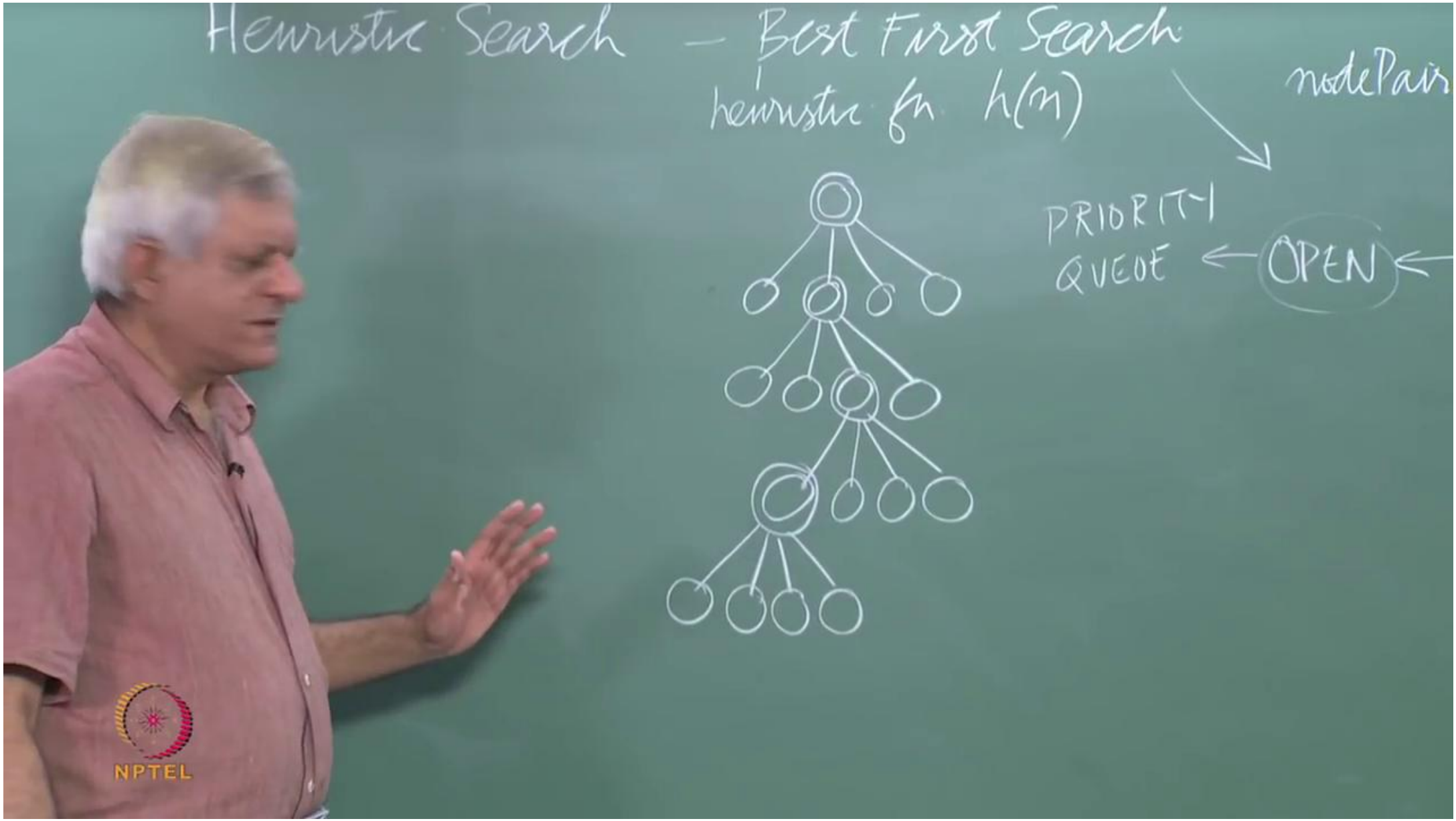

Best First Search
heuristic fn h(n)
nodePair ← Head(OPEN)
OPEN ← sort_h(append New Tail(OPEN))
NPTEL

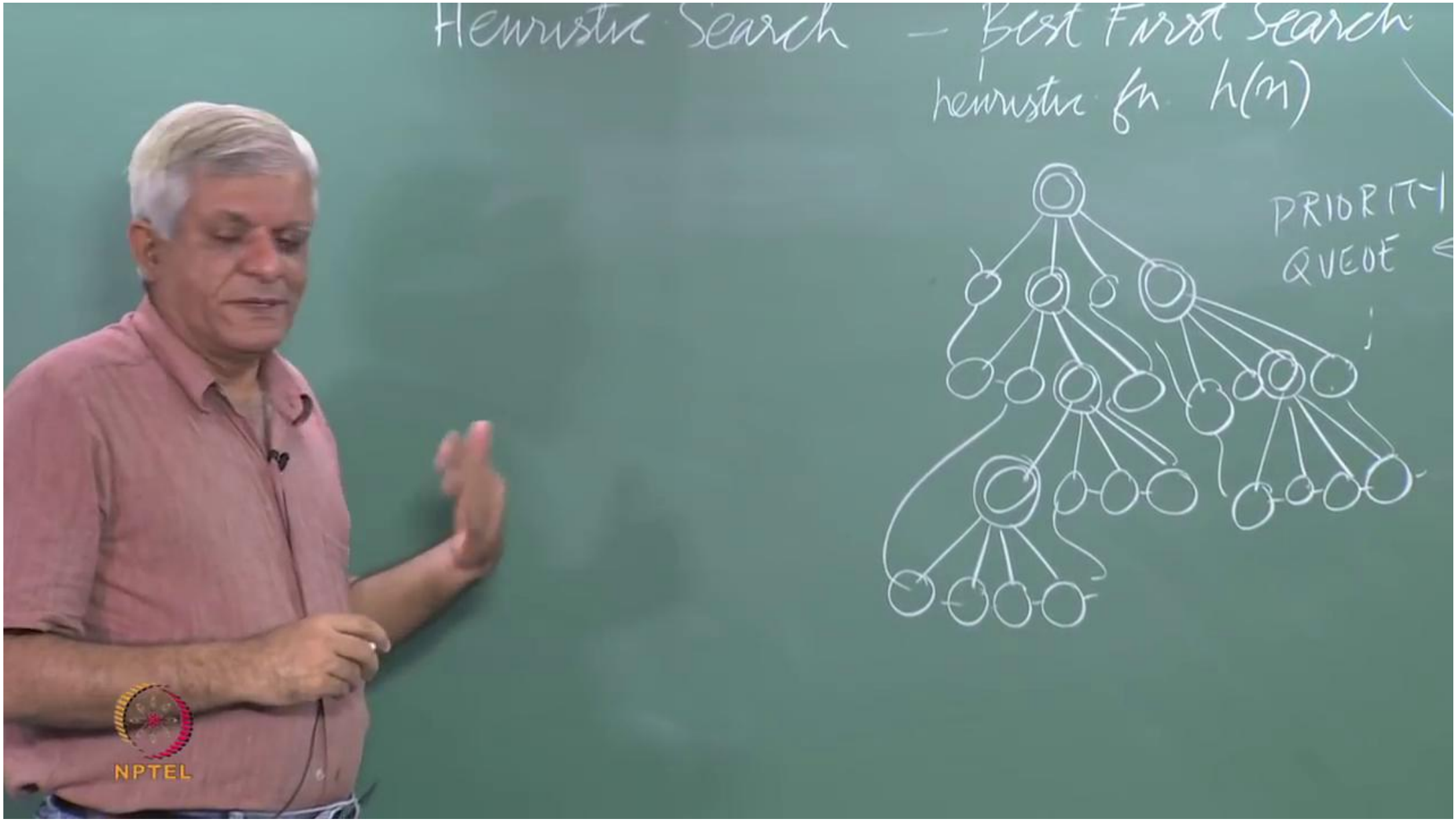

Best First Search
heuristic fn h(n)
nodePair ← Head(OPEN)
PRIORITY QUEUE ← OPEN ← sort_h(append New
NPTEL

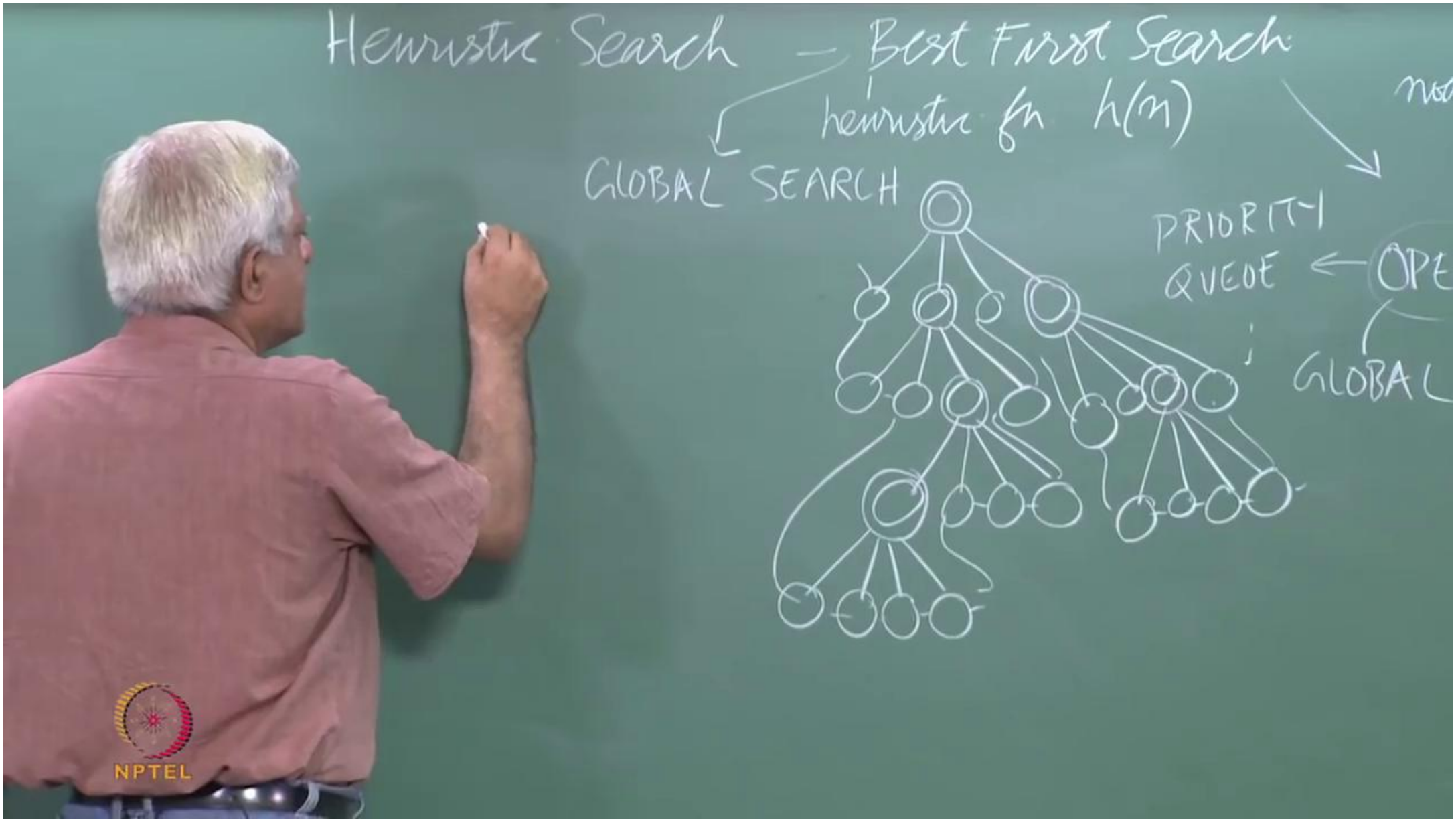

Heuristic Search – Best First Search
heuristic fn h(n)
nodePair
PRIORITY QUEUE
OPEN
NPTEL


Heuristic Search – Best First Search
heuristic fn h(n)
nodePair
PRIORITY QUEUE
OPEN
NPTEL


Heuristic Search – Best First Search
heuristic fn h(n)
nodePair
PRIORITY QUEUE
OPEN
NPTEL


Heuristic Search – Best First Search
heuristic fn h(n)
nodePair
PRIORITY QUEUE
OPEN
NPTEL


Heuristic Search – Best First Search
heuristic fn h(n)
PRIORITY
QUEUE

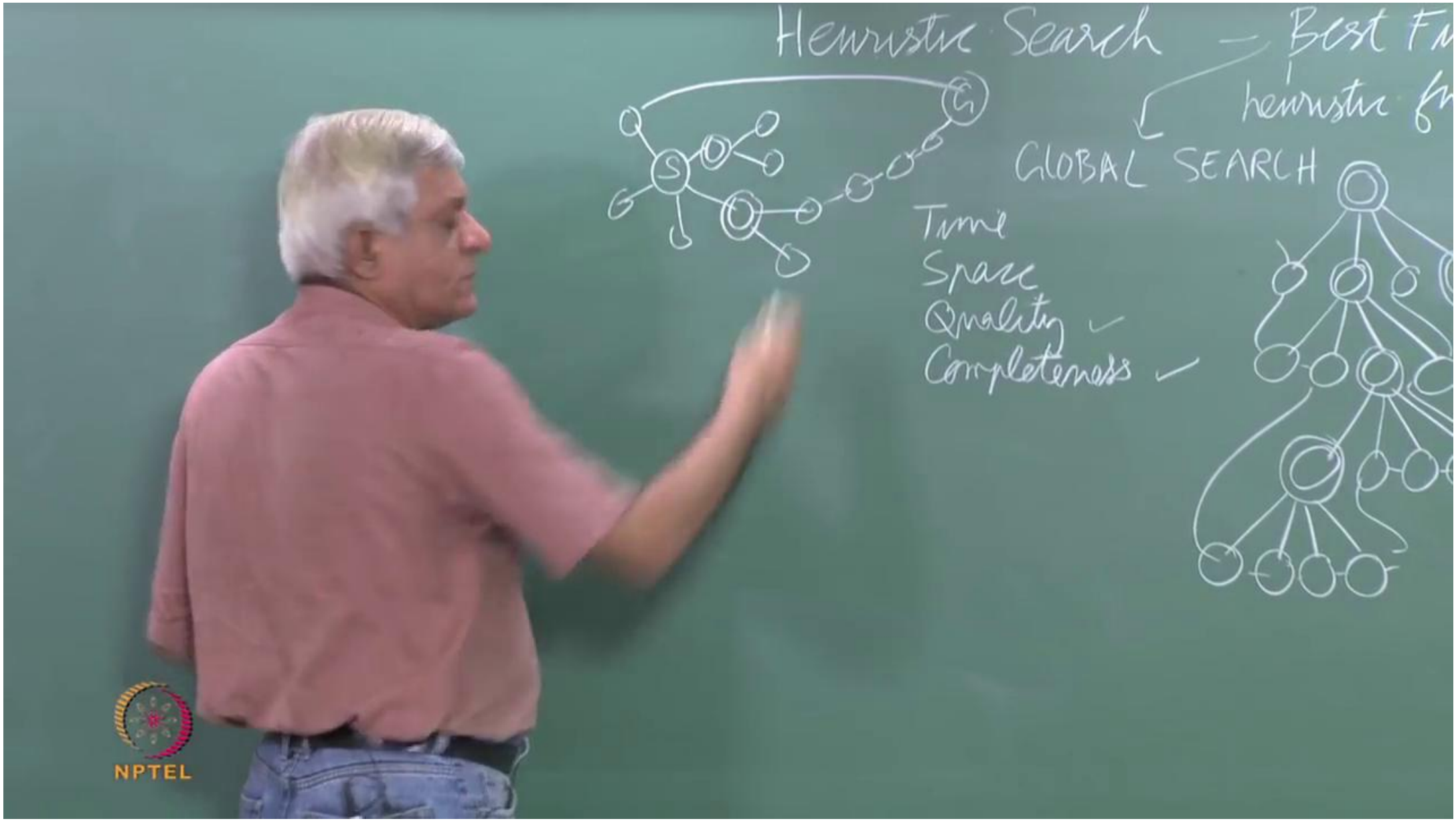

Heuristic Search — Best First Search
heuristic fn h(n)
GLOBAL SEARCH
PRIORITY
QUEUE
OPE
GLOBAL
NPTEL

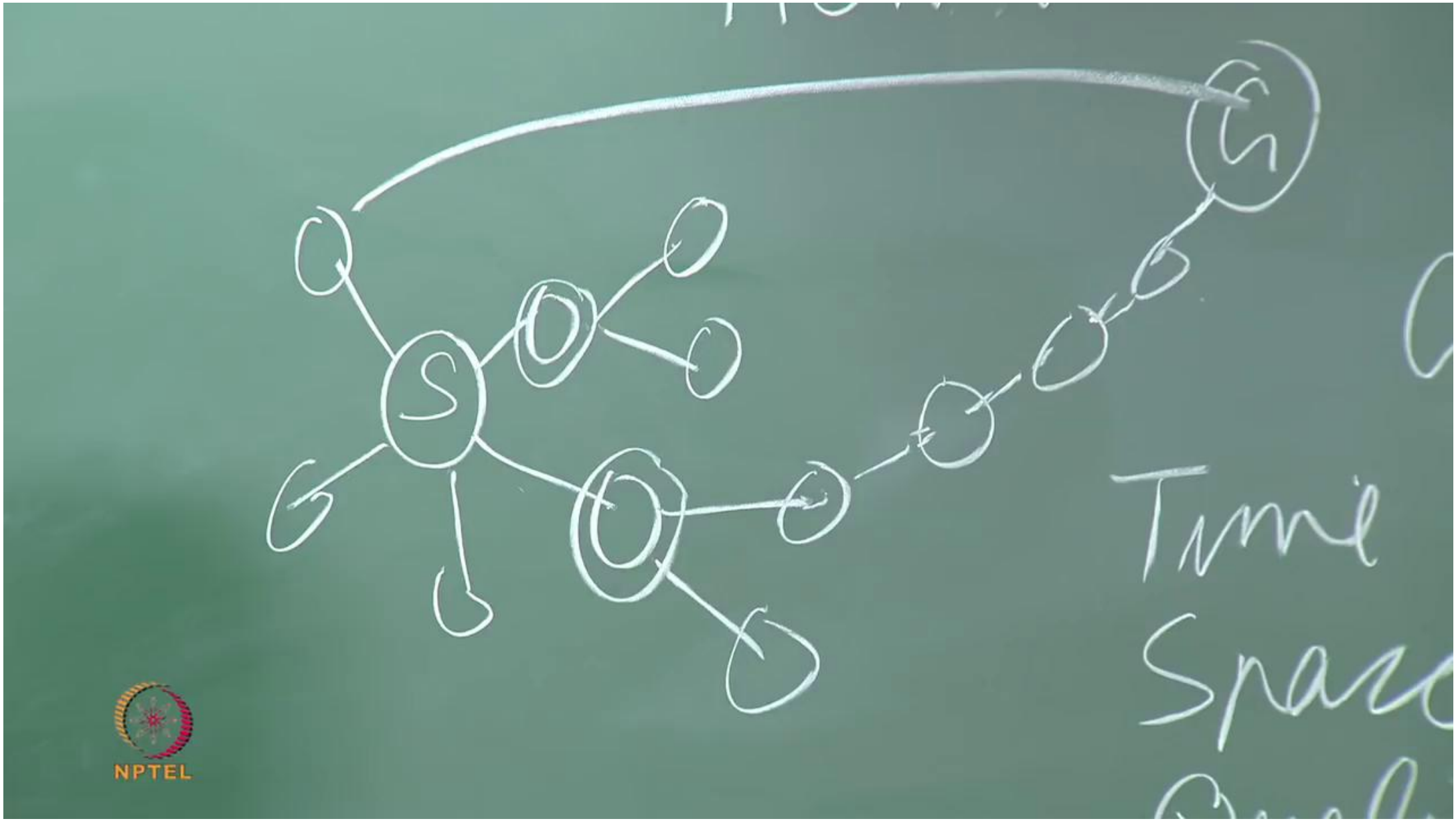
Time

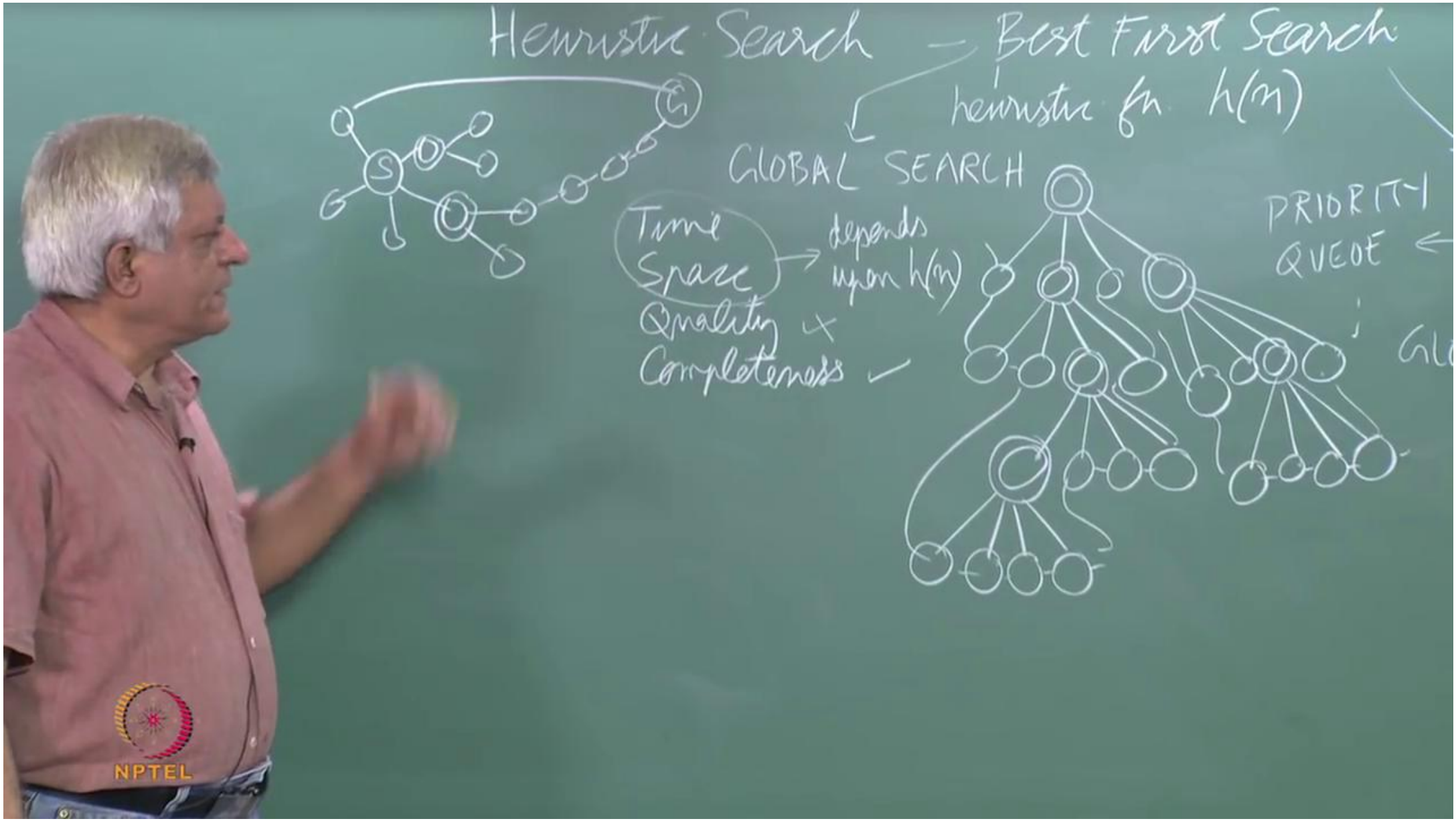
Space

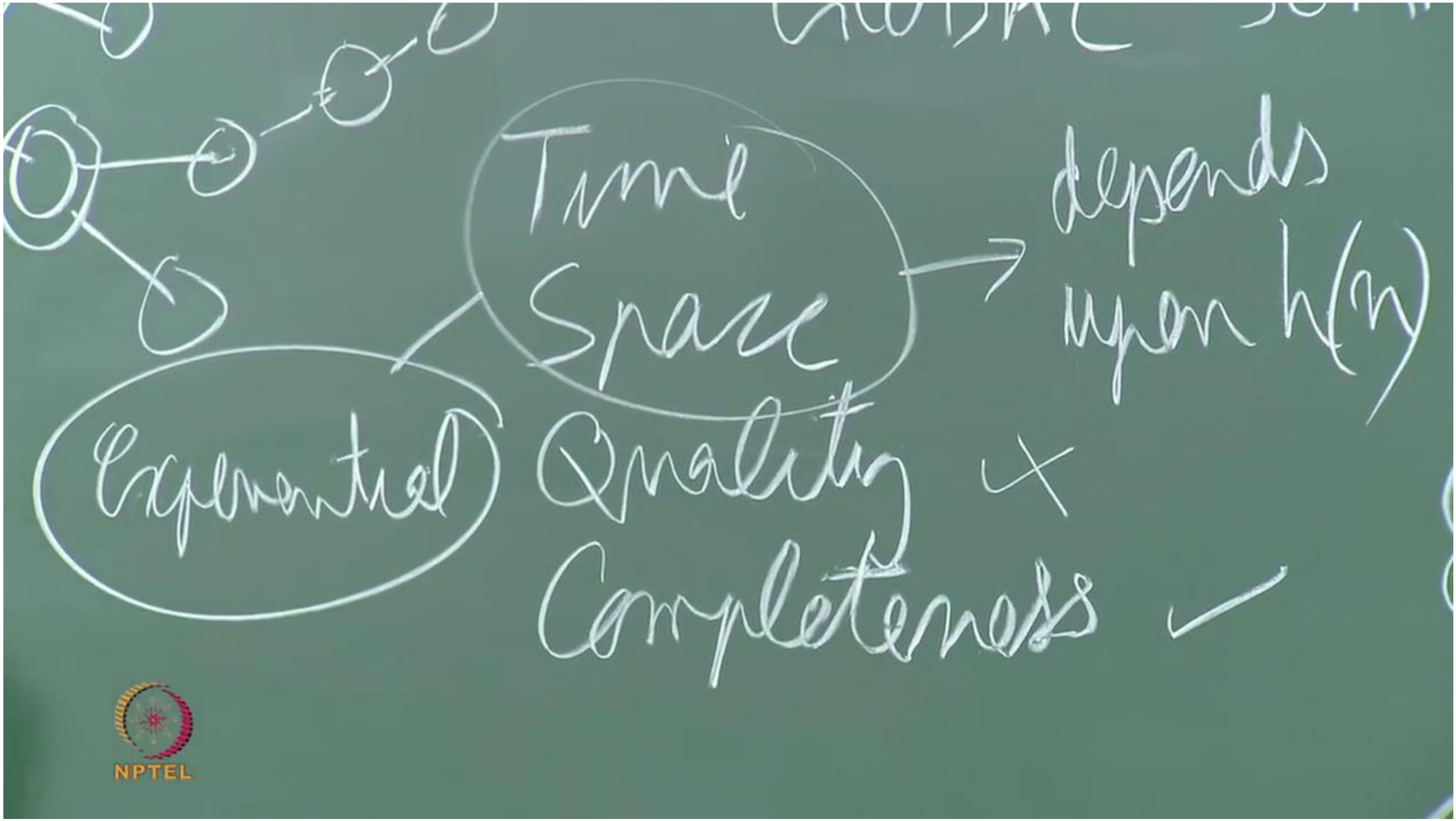
Quality

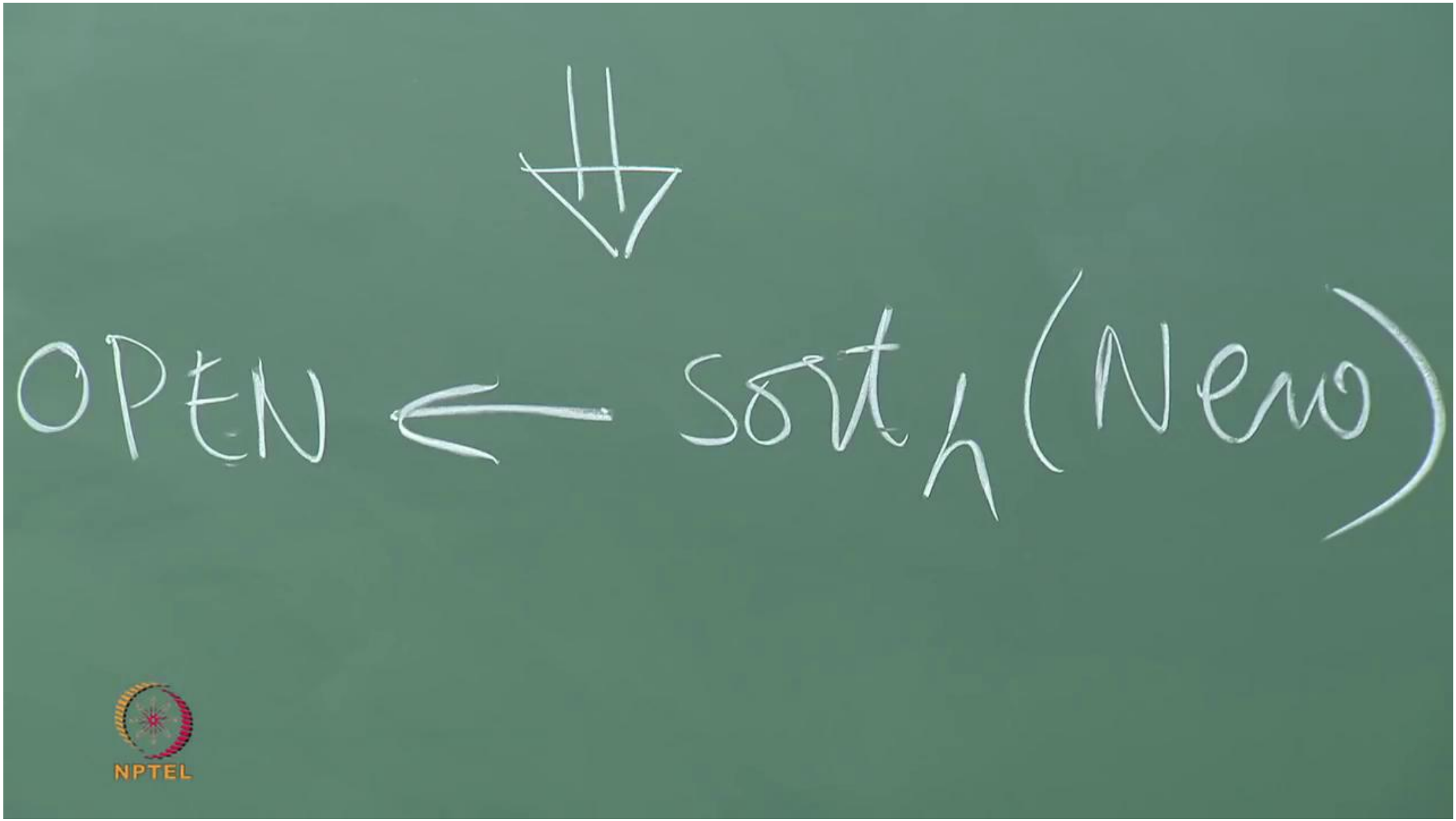
Completeness

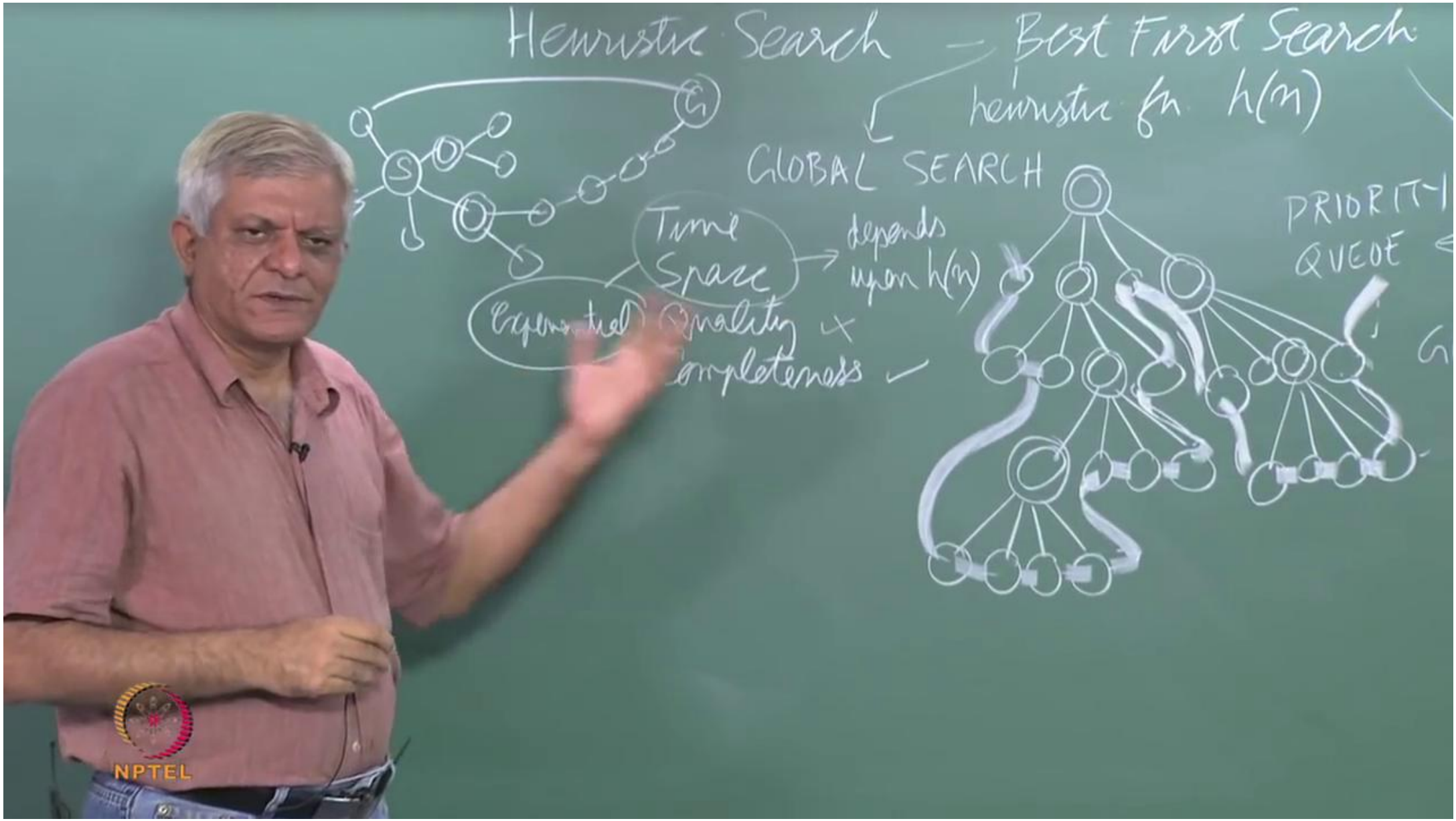

Heuristic Search
Best Fi
heuristic fu
GLOBAL SEARCH
Time
Space
Quality
Completeness
S
NPTEL

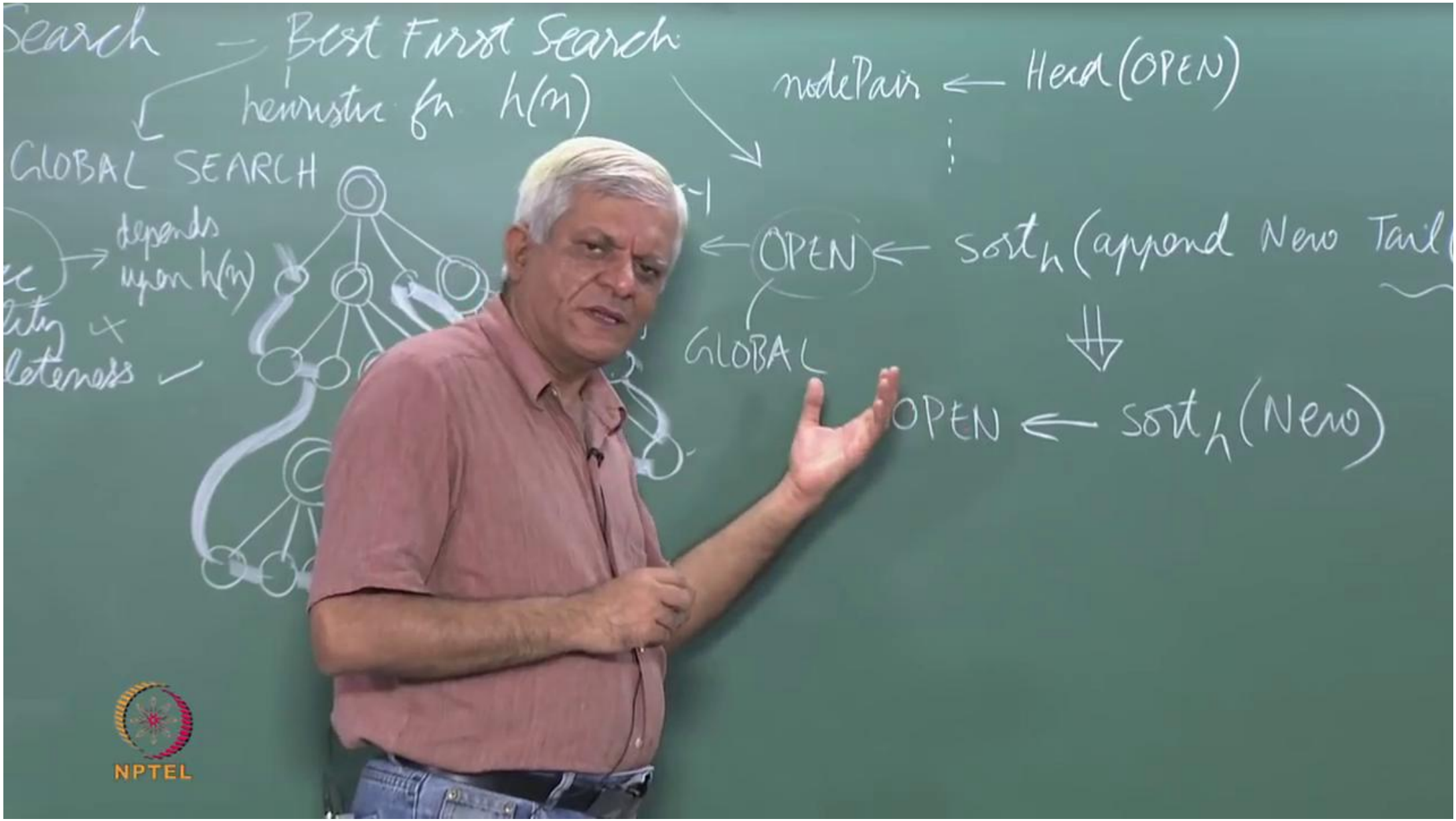

S
Time
Spac

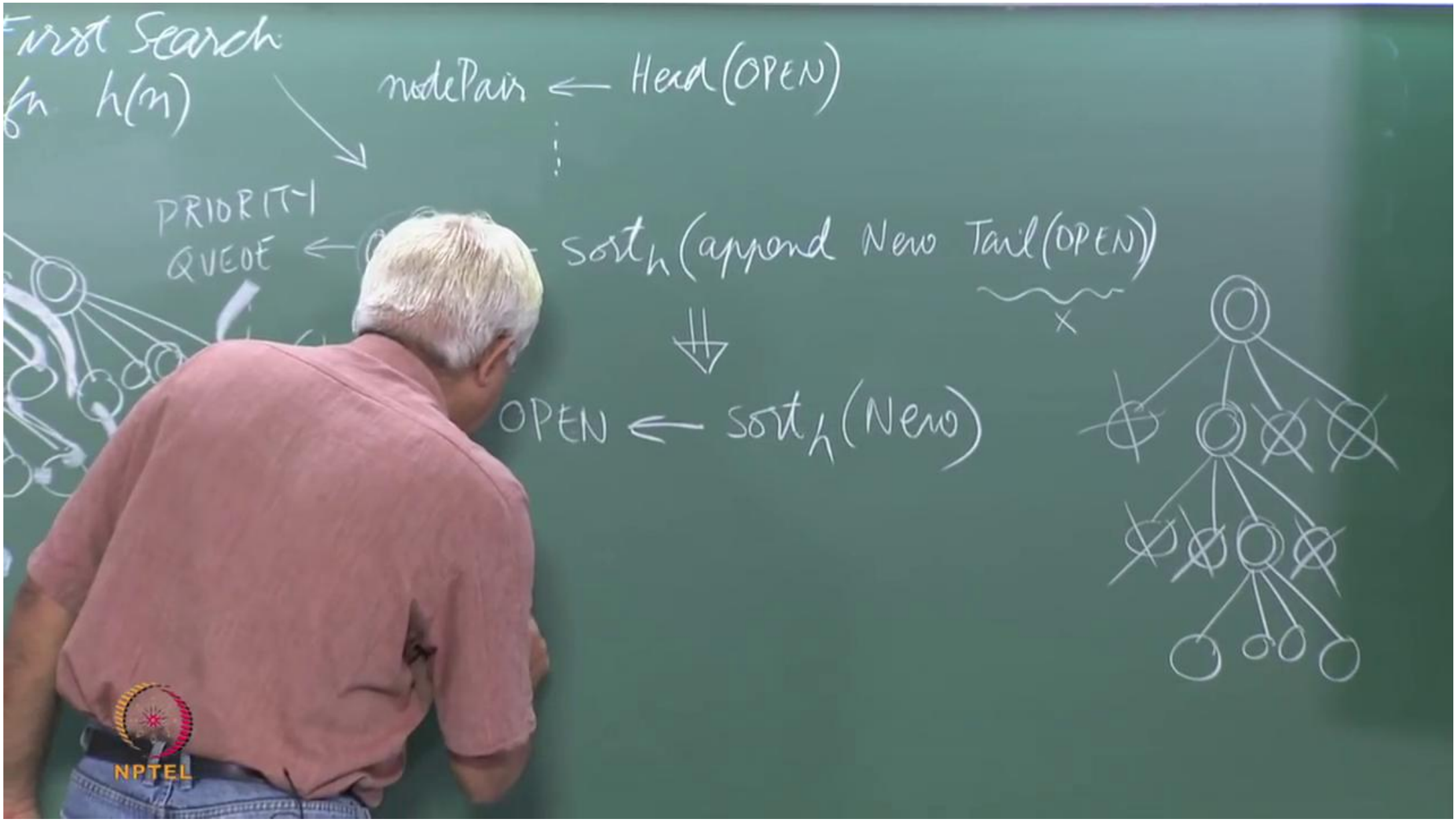

Heuristic Search — Best First Search
heuristic fn h(n)
GLOBAL SEARCH
Time
Space
depends upon h(n)
Quality +
Completeness —
PRIORITY QUEUE
NPTEL

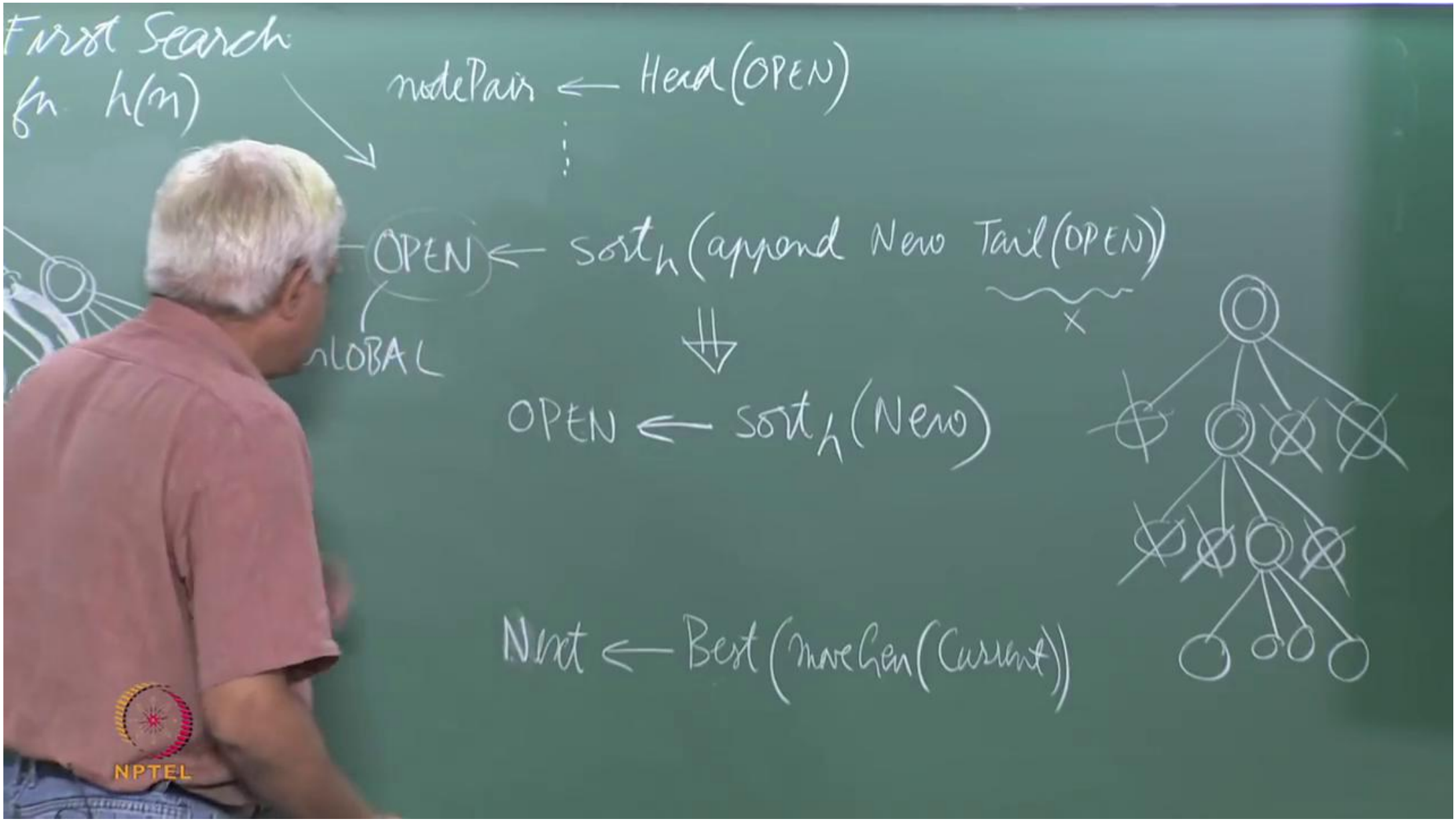

Time
Space
depends upon h(n)
Exponential
Quality +
Completeness

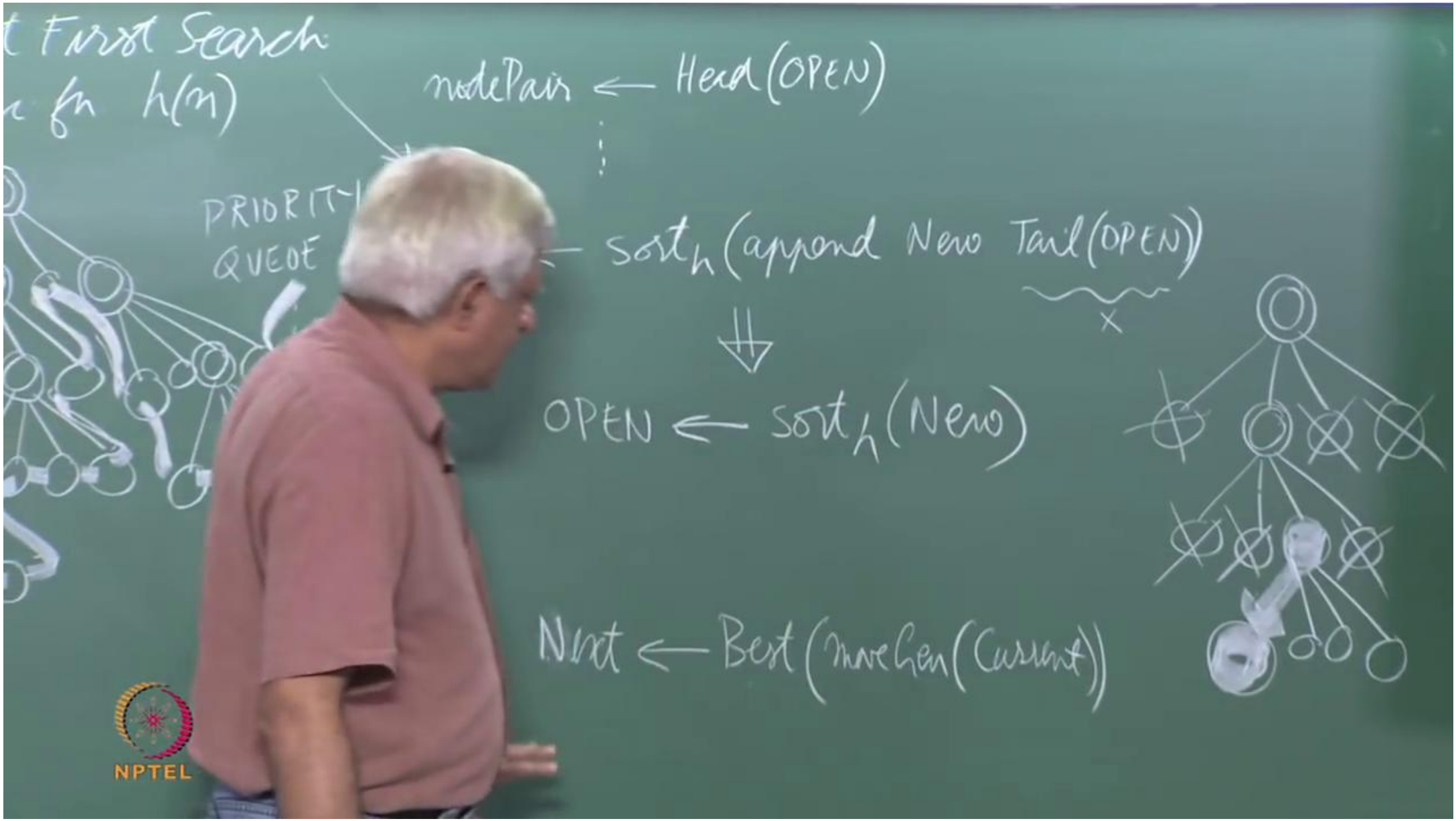

OPEN ← sort_h(New)

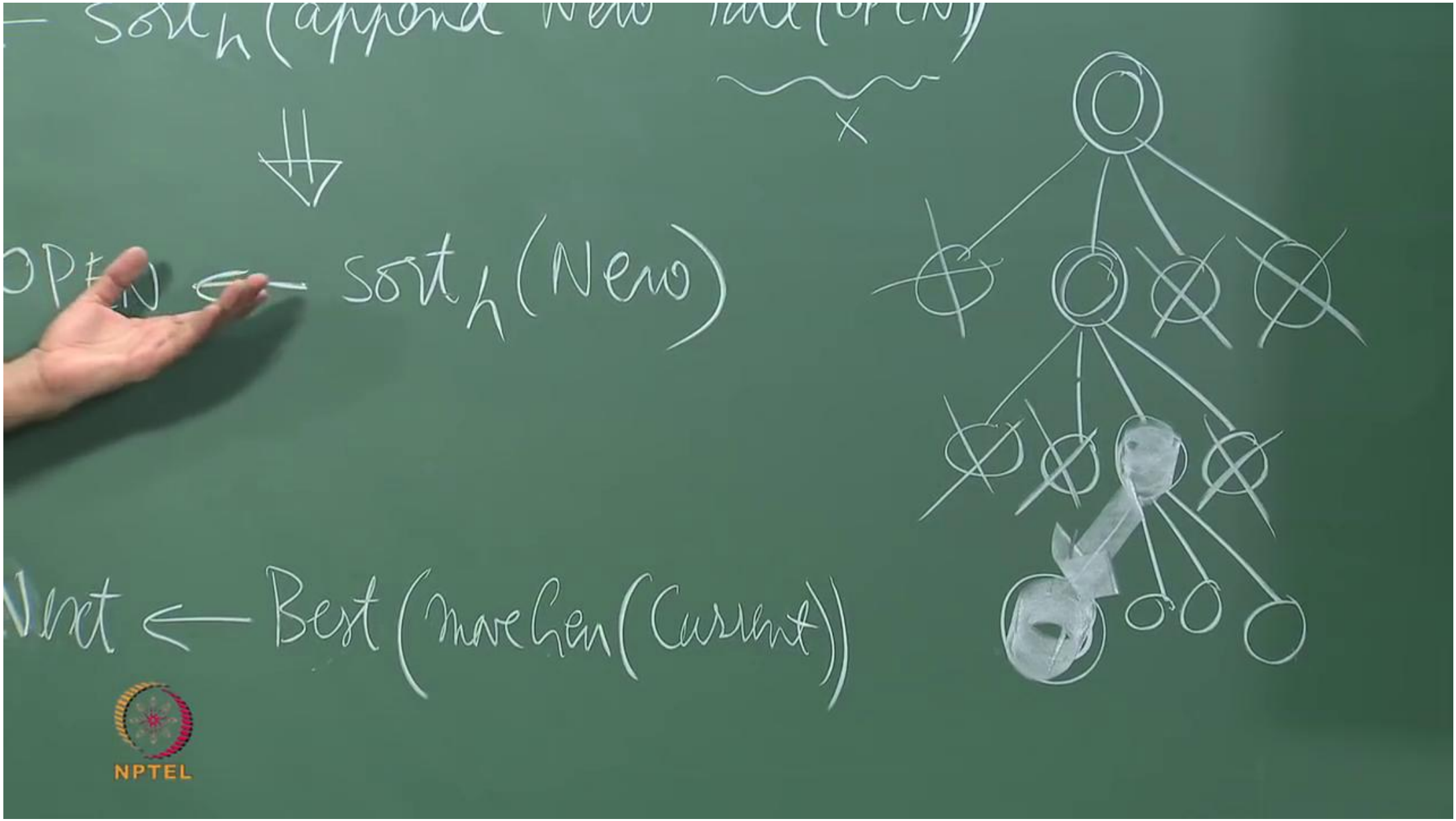

Heuristic Search — Best First Search
heuristic fn h(n)
GLOBAL SEARCH
Time
Space
depends upon h(n)
Exponential
Quality +
completeness —
PRIORITY QUEUE
NPTEL


Search — Best First Search
heuristic fn h(n)
GLOBAL SEARCH
depends upon h(n)
nodePair ← Head(OPEN)
OPEN ← sort_h(append New Tail
GLOBAL
OPEN ← sort_h(New)
NPTEL


First Search
fn h(n)
nodePair ← Head(OPEN)
PRIORITY
QUEUE
sort_h(append New Tail(OPEN))
OPEN ← sort_h(New)


First Search
fn h(n)
nodePair ← Head(OPEN)
OPEN ← sort_h(append New Tail(OPEN))
OPEN ← sort_h(New)
Next ← Best(moveGen(Current))
NPTEL

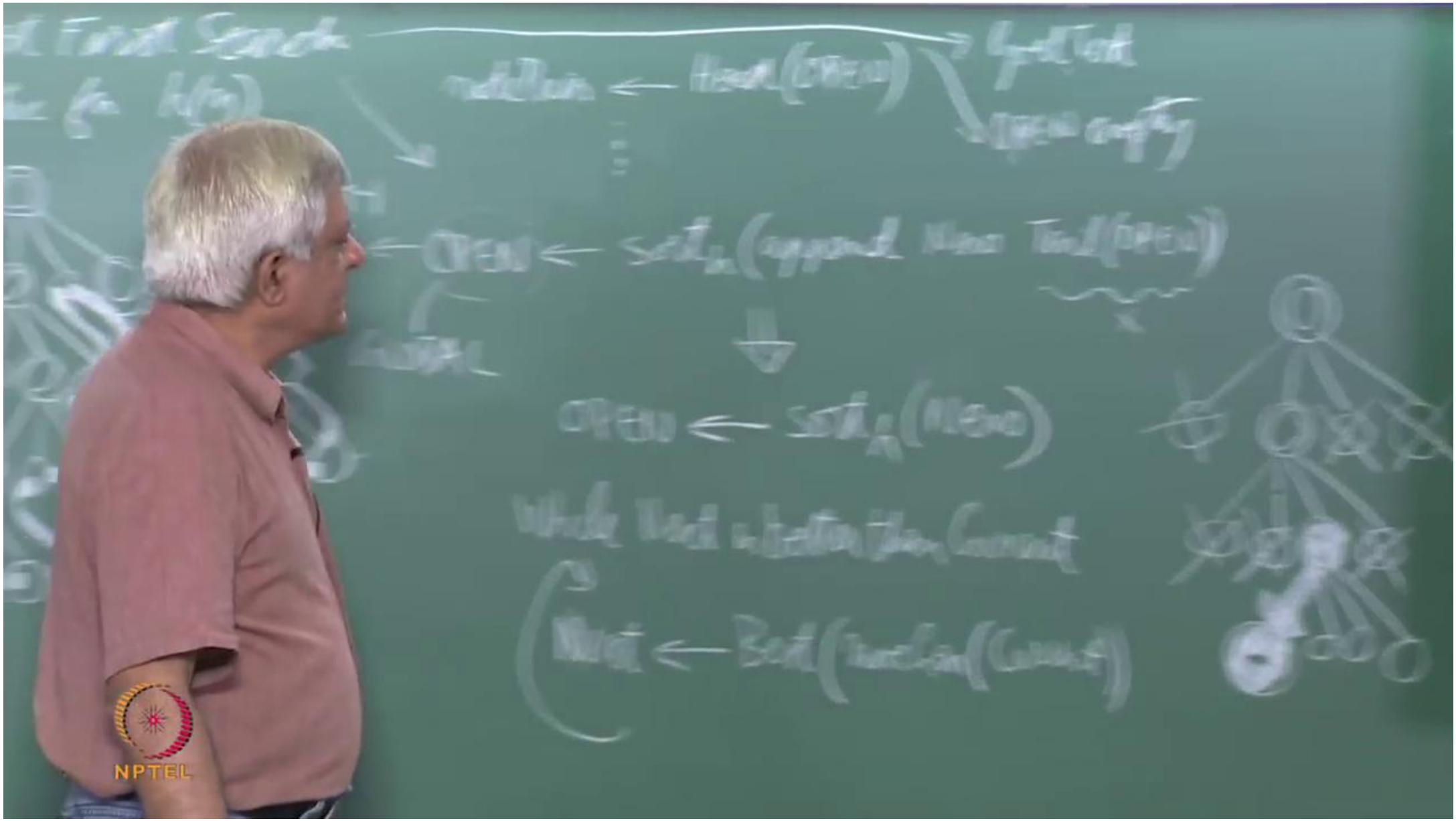

First Search
h(n)
nodePair ← Head(OPEN)
PRIORITY
QUEUE
sort_h(append New Tail(OPEN))
OPEN ← sort_h(New)
Next ← Best(moveGen(Current))
NPTEL

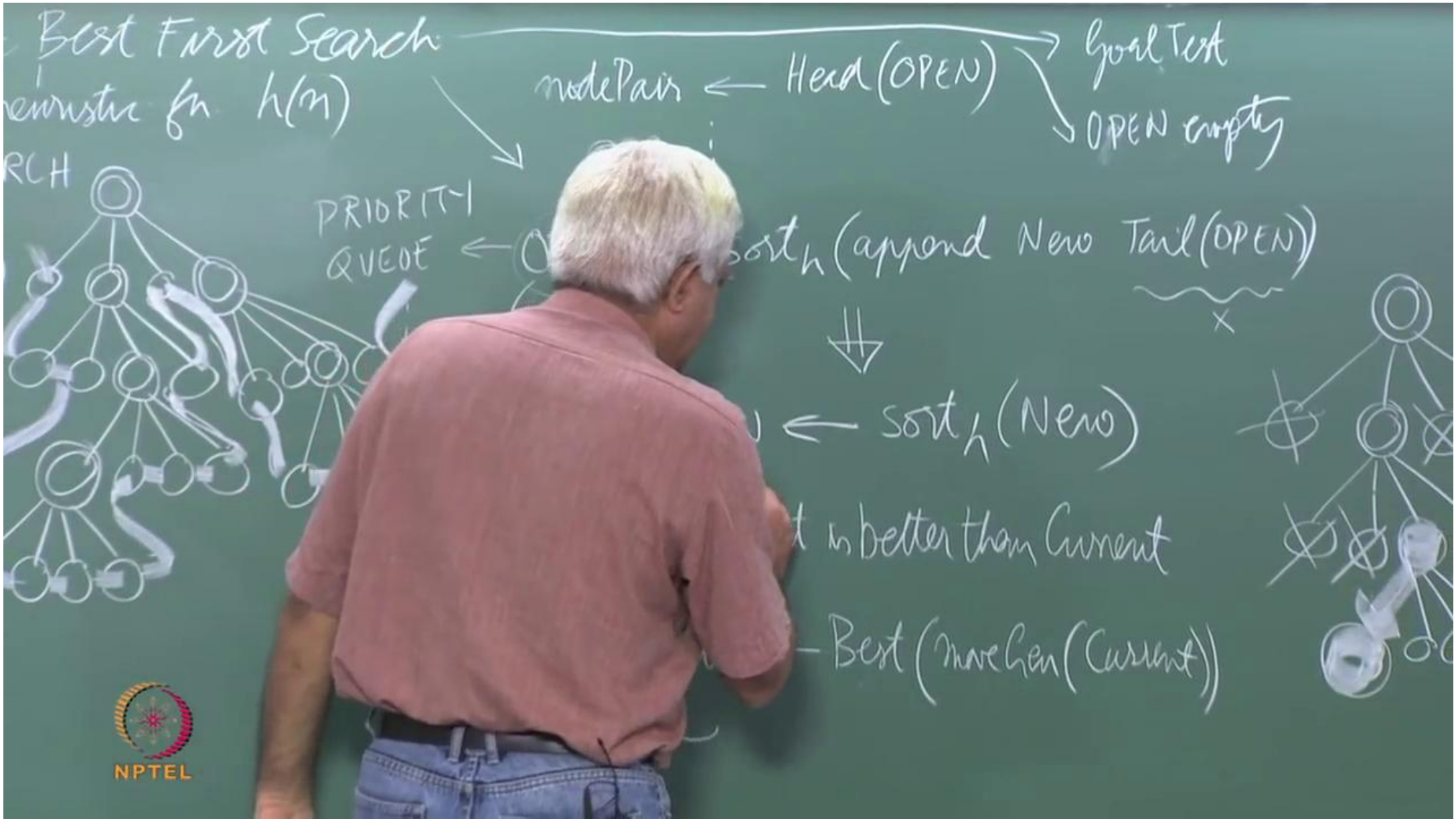

sort_h(append New
OPEN ← sort_h(New)
Next ← Best(moveGen(Current))
NPTEL

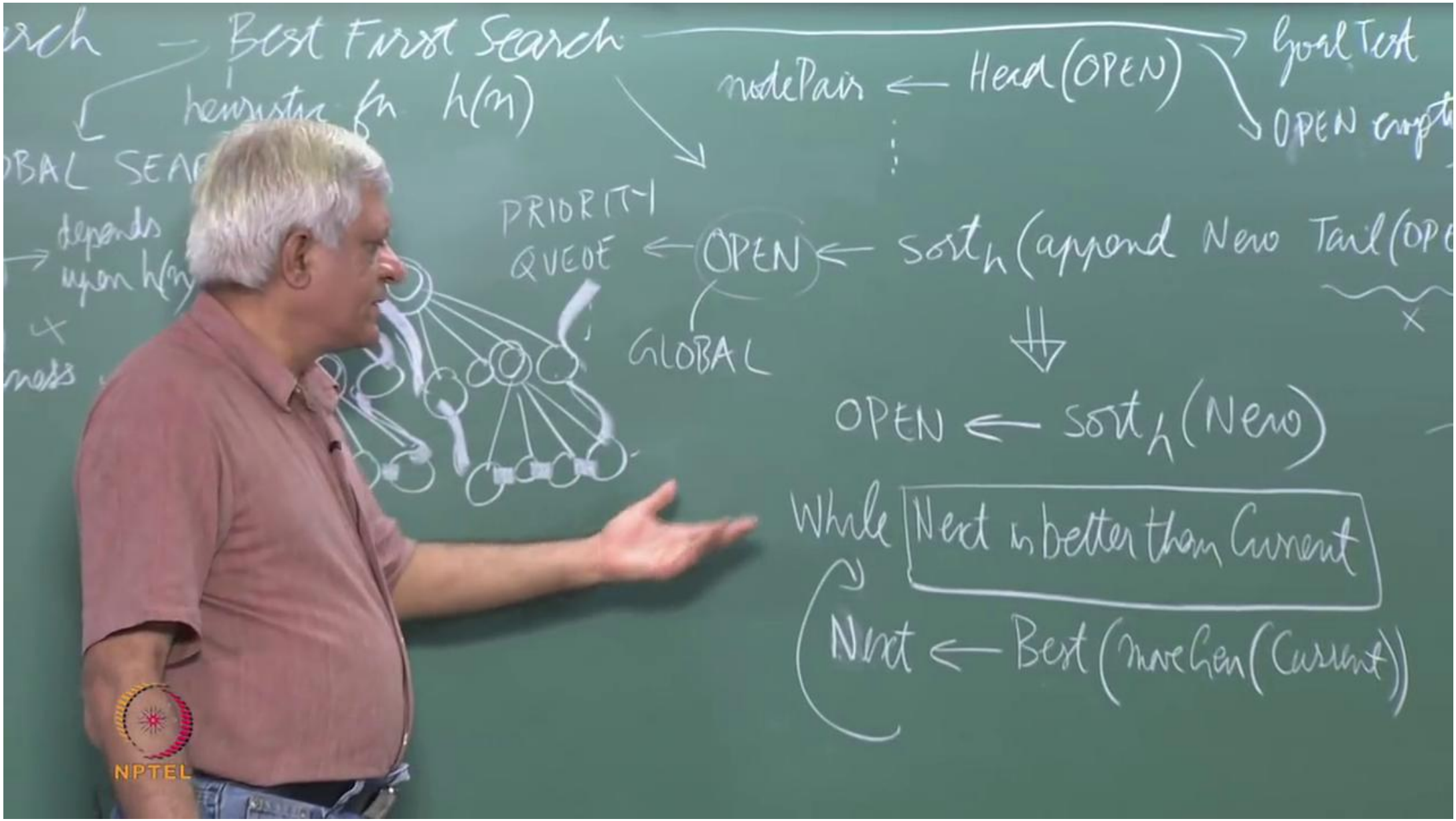
OPEN ← $sort_h$(New)

While Next is better than Current

Next ← Best(moveGen(Current))

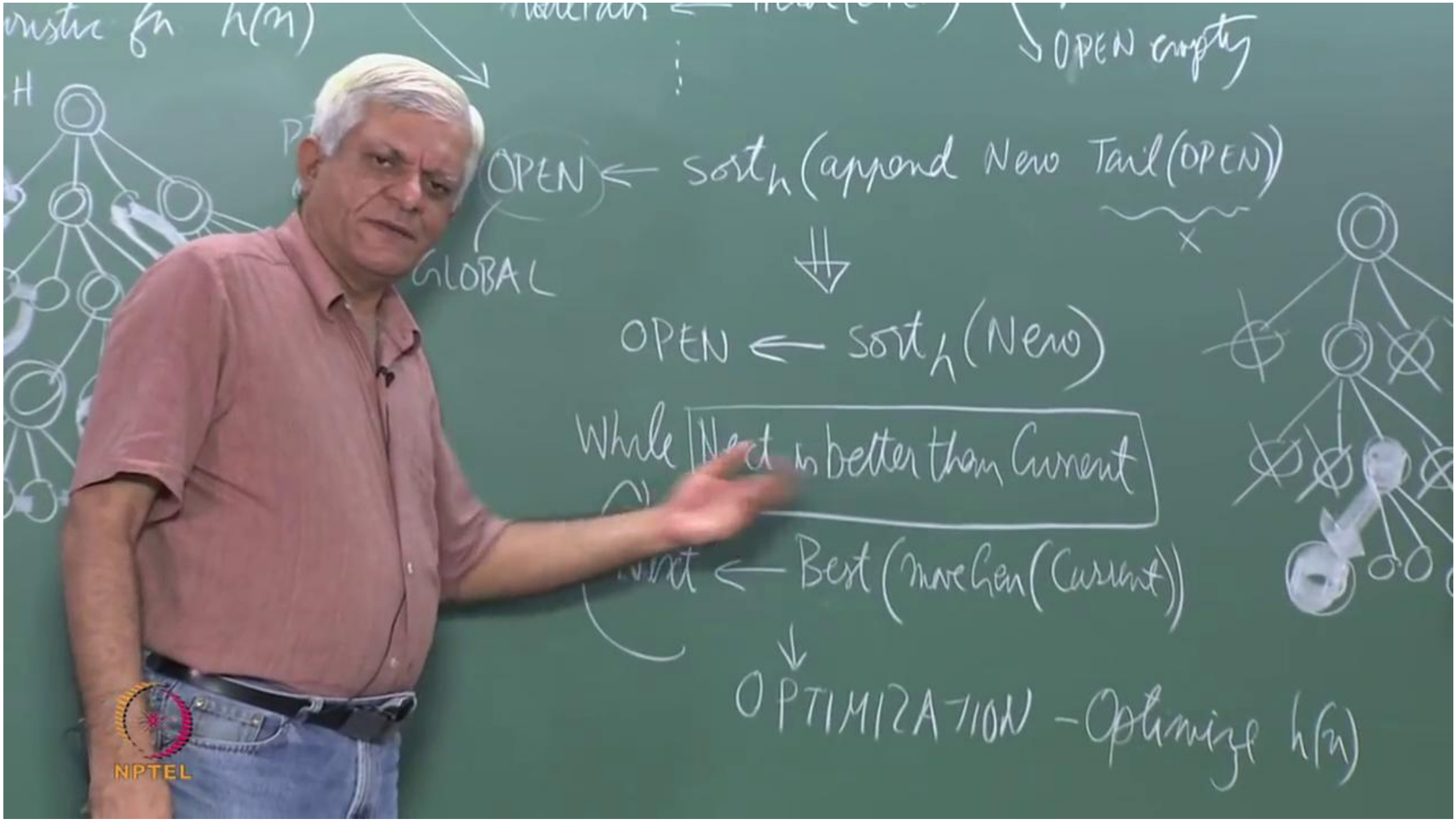

NPTEL

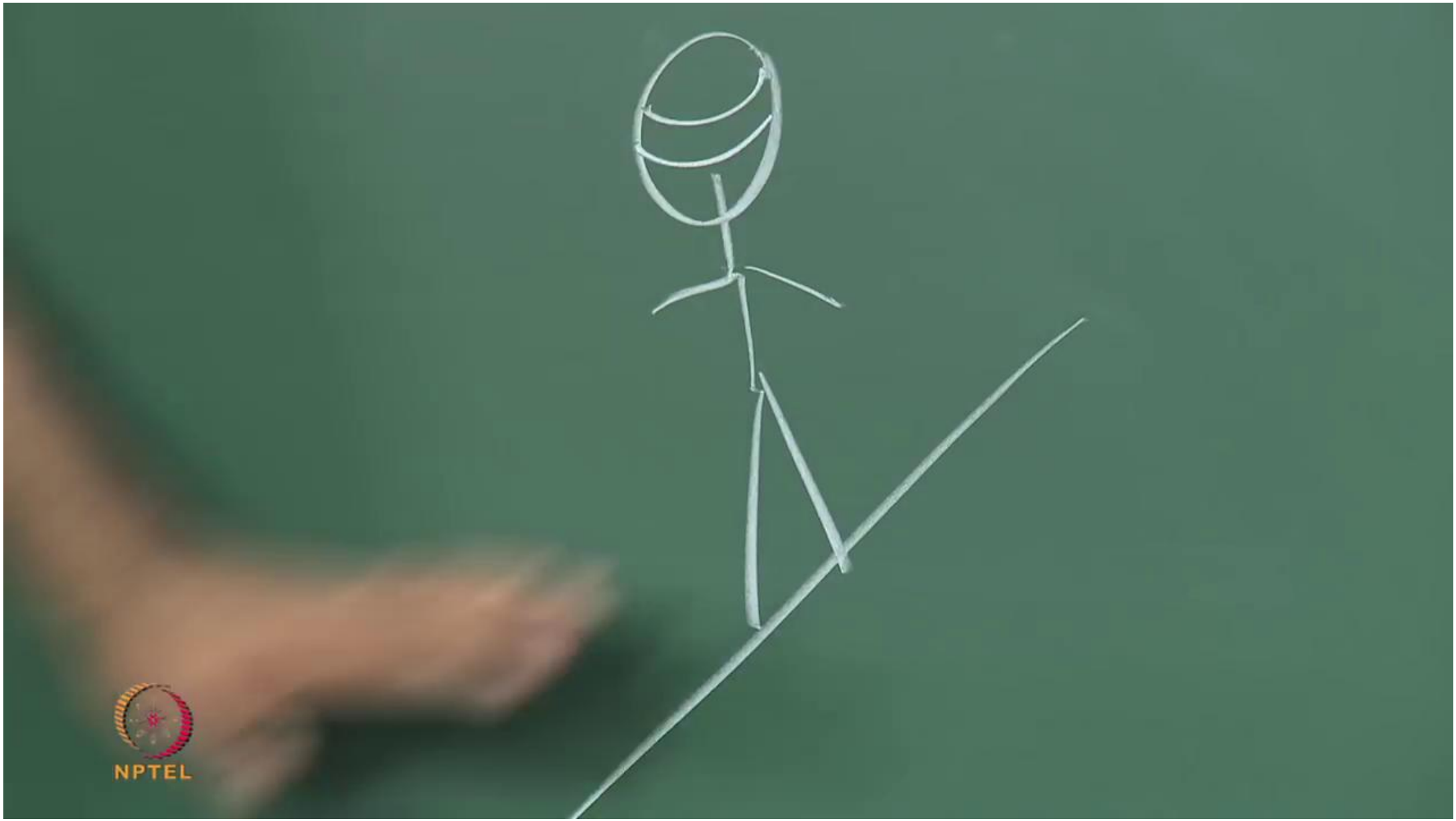

Best First Search
nodePair ← Head(OPEN)
Goal Test
OPEN empty
PRIORITY
QUEUE
append New Tail(OPEN))
sort
sort (New)
is better than Current
Best (moveGen(Current))
NPTEL


Best First Search
heuristic fn h(n)
nodePair ← Head(OPEN)
GoalTest
OPEN empty
PRIORITY QUEUE
OPEN
GLOBAL
sort_h(append New Tail(OPEN
OPEN ← sort_h(New)
While Next is better than Current
Next ← Best(moveGen(Current))
depends upon h(n)
NPTEL

While Next is better than Current

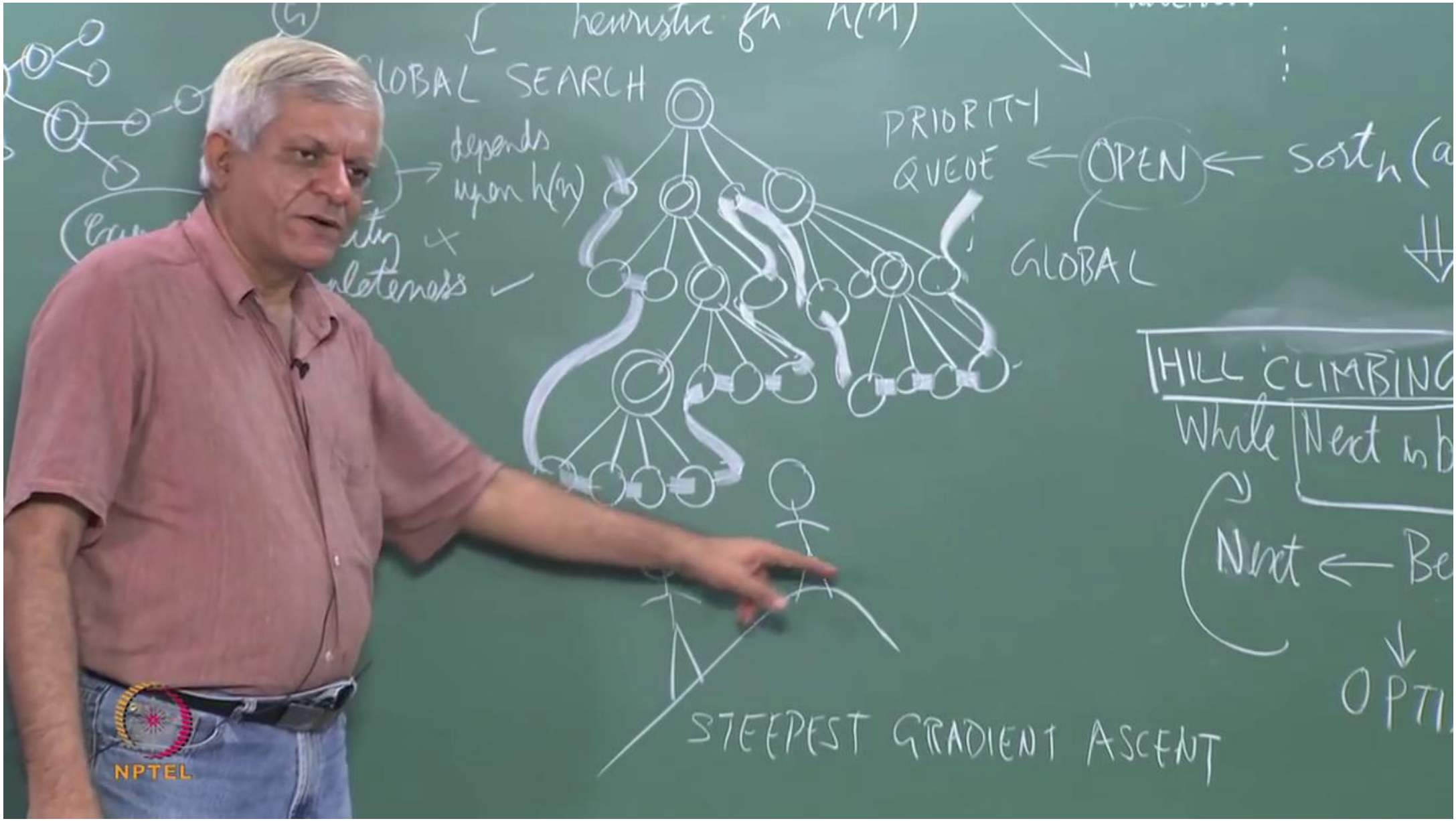
Next ← Best (moveGen (Current))

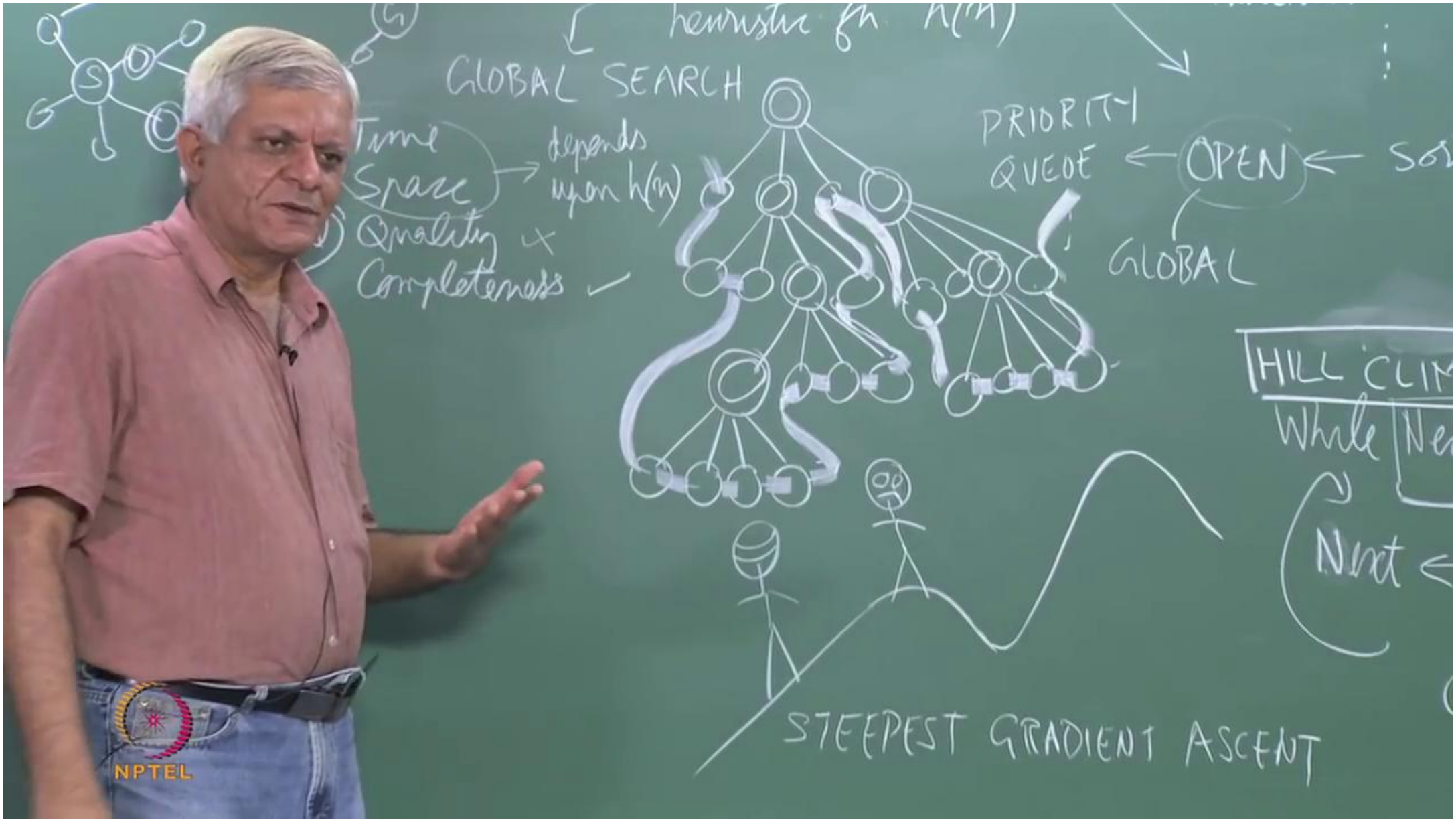

OPEN empty
OPEN ← sort_h (append New Tail (OPEN))
GLOBAL
OPEN ← sort_h (New)
While Next is better than Current
Next ← Best (moveGen (Current))
OPTIMIZATION - Optimize h(n)
NPTEL

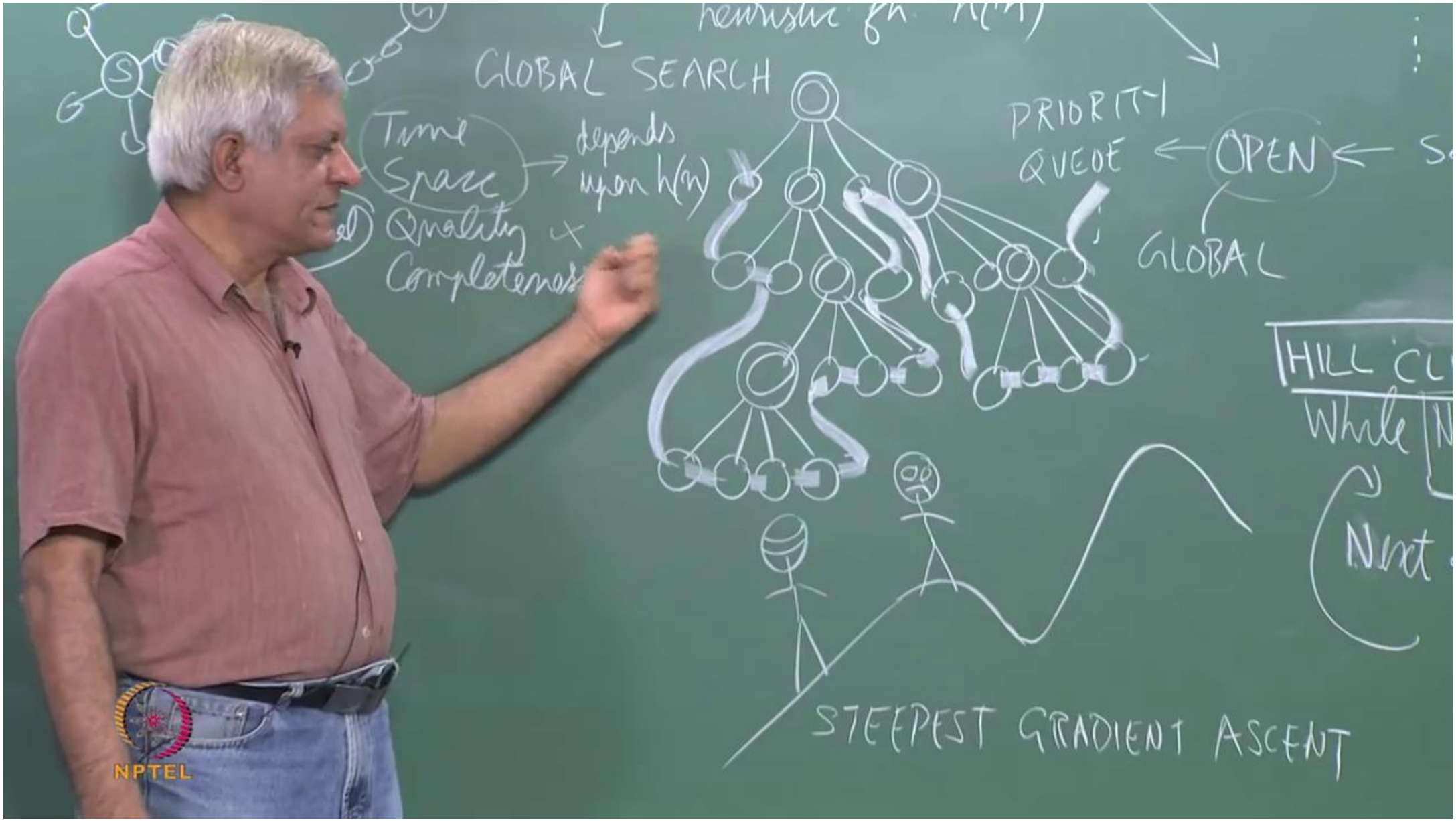

NPTEL

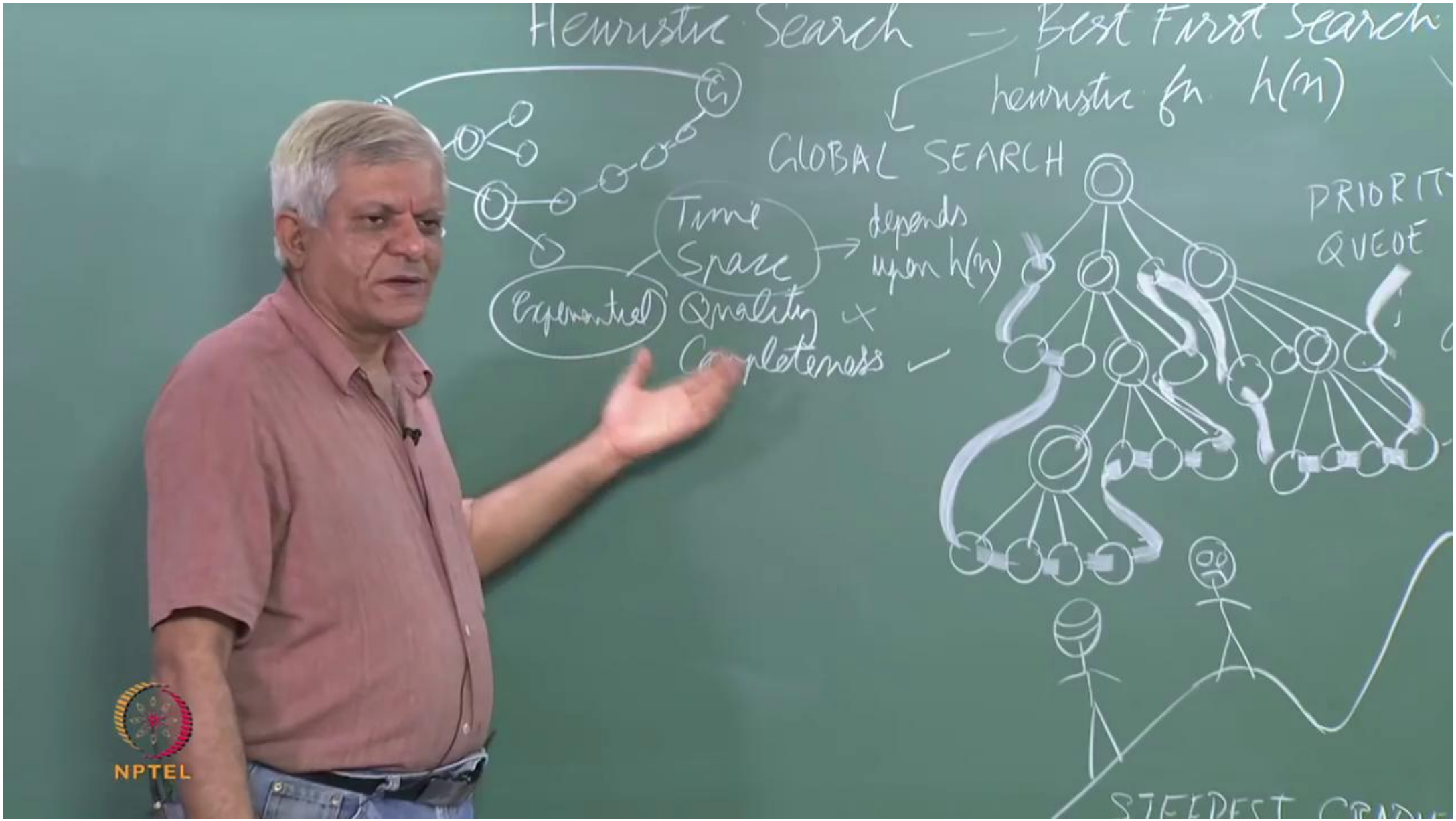

STEEPEST GRADIENT ASCENT
NPTEL

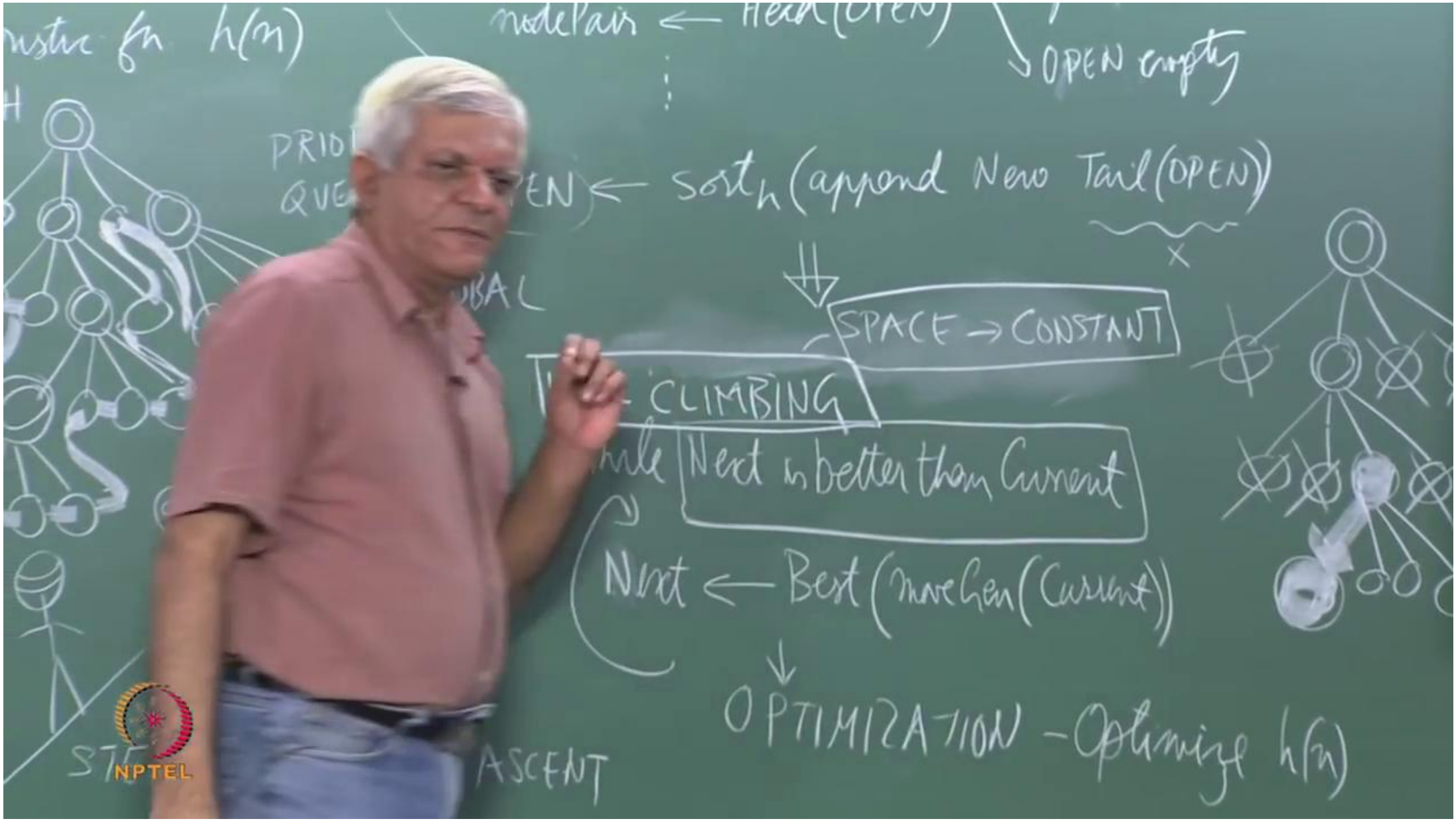

STEEPEST GRADIENT ASCENT
NPTEL

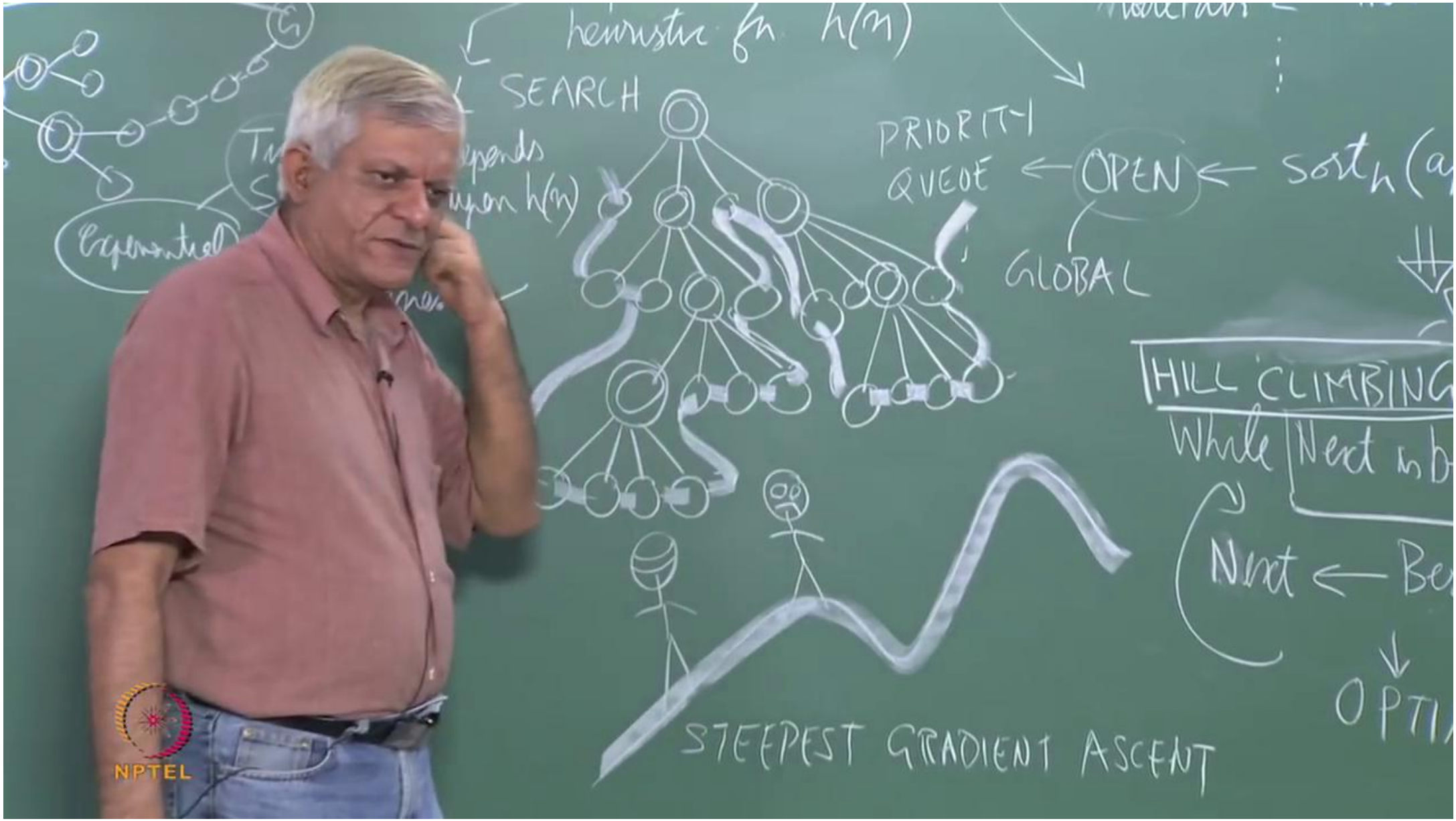

heuristic fn h(n)
GLOBAL SEARCH
depends upon h(n)
PRIORITY QUEUE
OPEN
GLOBAL
HILL CLIMBING
While
Next
STEEPEST GRADIENT ASCENT
NPTEL


GLOBAL SEARCH
Time
Space
Quality
Completeness
depends upon h(n)
PRIORITY QUEUE
OPEN
GLOBAL
HILL CLIM
While
Next
STEEPEST GRADIENT ASCENT
NPTEL


GLOBAL SEARCH
Time
Space
depends upon h(n)
Quality
Completeness
PRIORITY QUEUE
OPEN
GLOBAL
HILL CL
While
Next
STEEPEST GRADIENT ASCENT
NPTEL

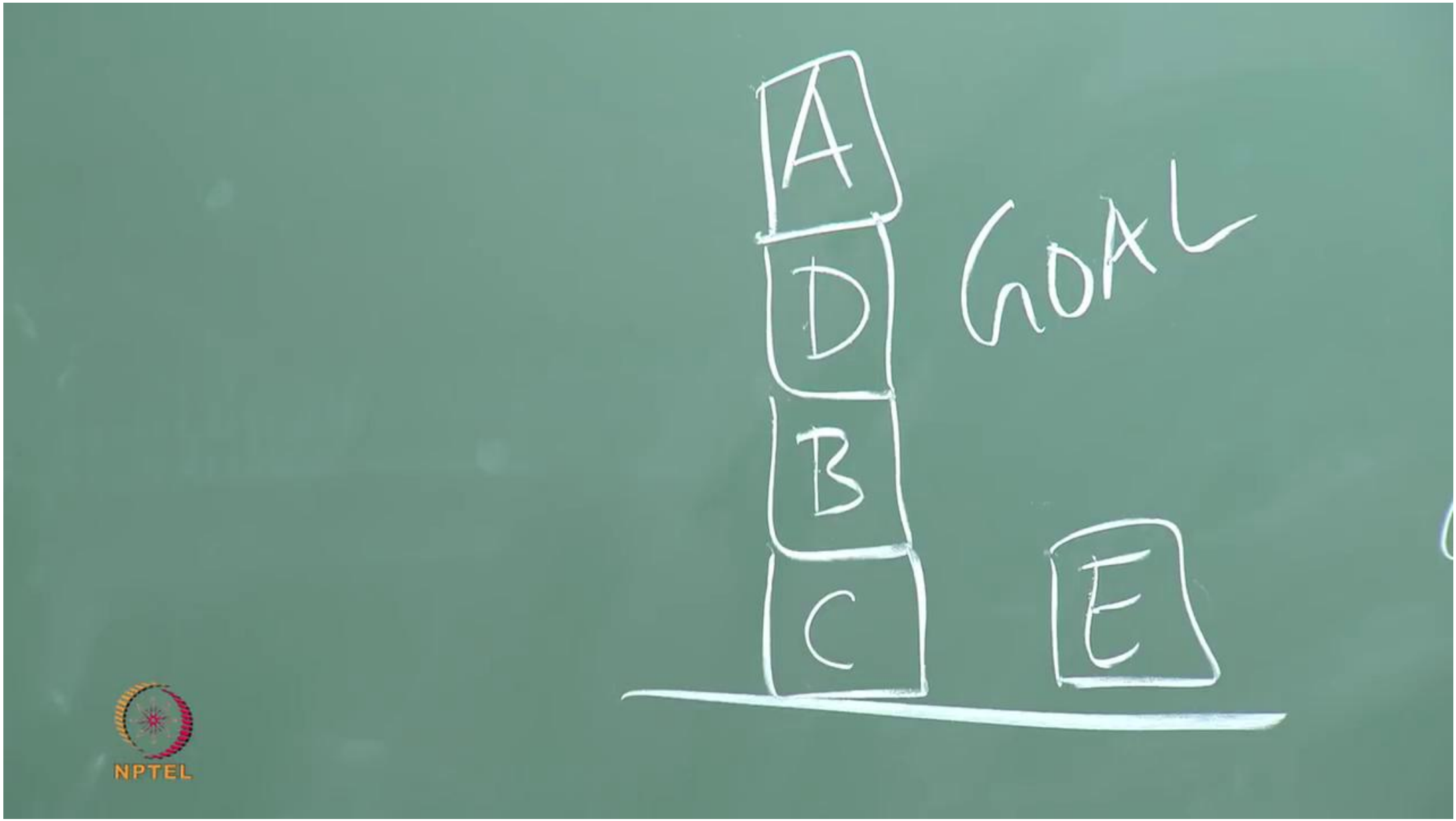

Heuristic Search — Best First Search
heuristic fn h(n)
GLOBAL SEARCH
Time
Space
depends upon h(n)
Exponential
Quality +
Completeness —
PRIORITY QUEUE
NPTEL

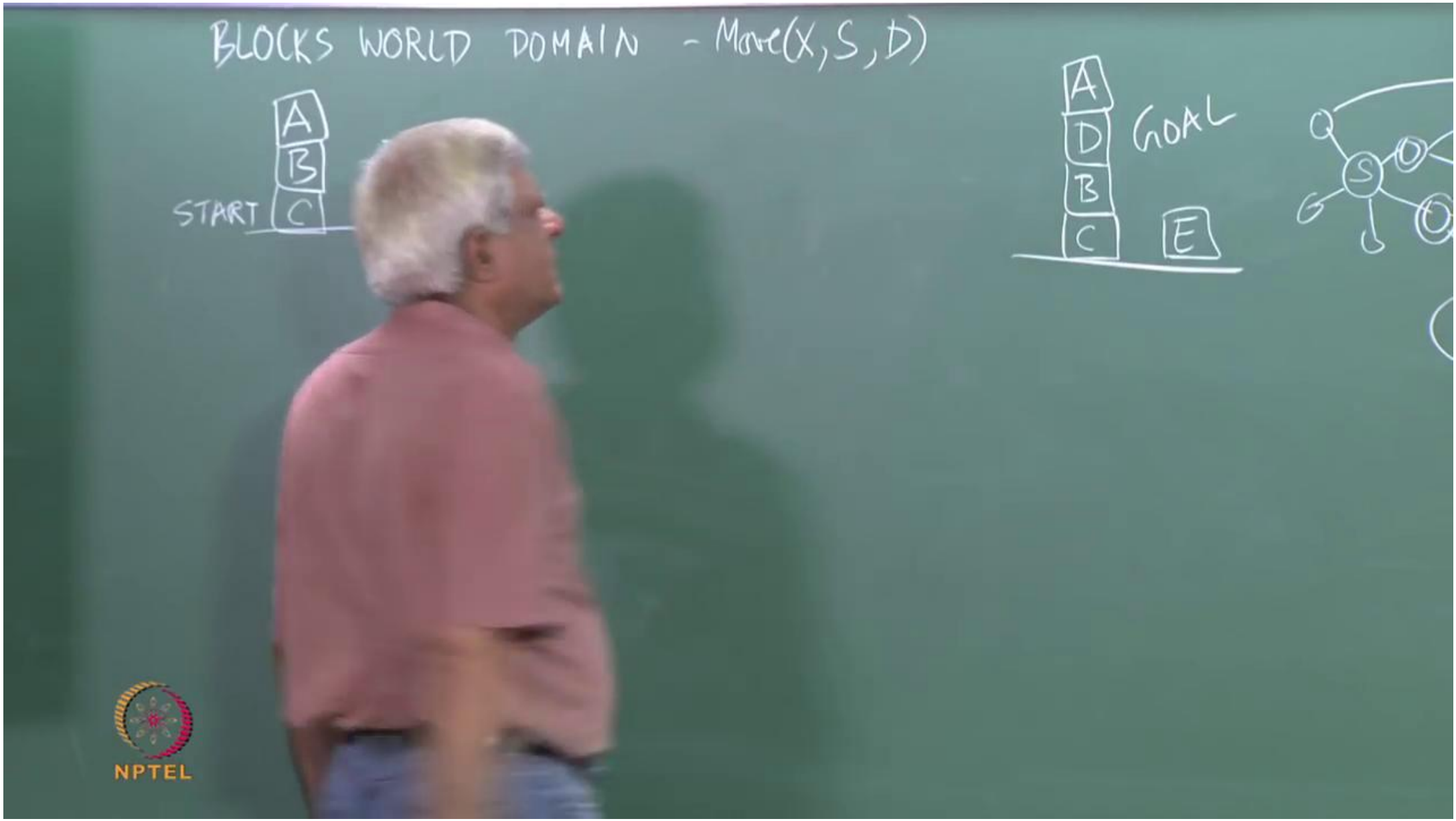

nodePair ← Head(OPEN)
OPEN empty
sort_h(append New Tail(OPEN))
SPACE → CONSTANT
CLIMBING
Next is better than Current
Next ← Best(moveGen(Current))
OPTIMIZATION - Optimize h(n)
ASCENT
NPTEL

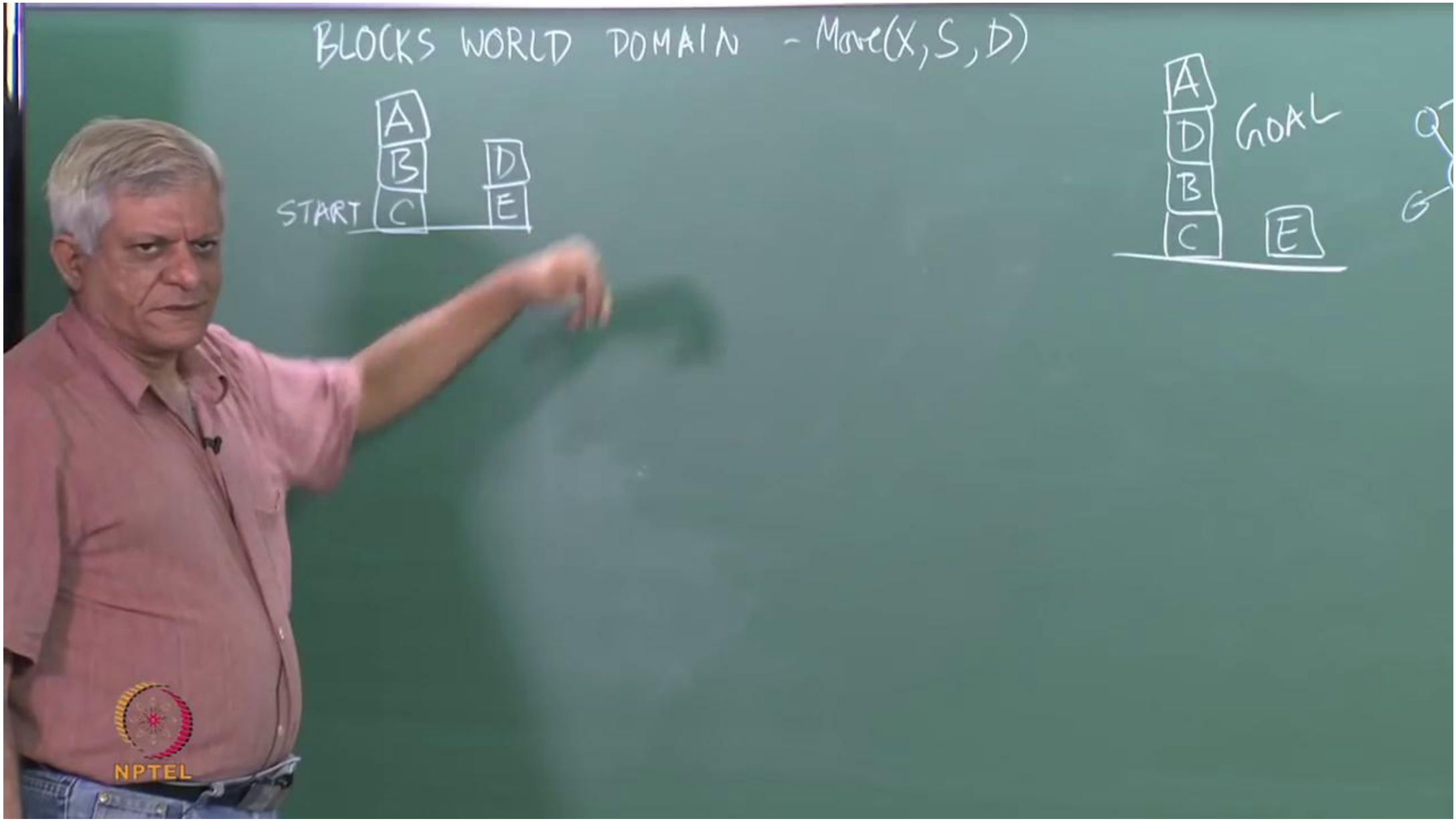

heuristic fn h(n)
SEARCH
PRIORITY QUEUE
OPEN
GLOBAL
HILL CLIMBING
While
Next
STEEPEST GRADIENT ASCENT
NPTEL

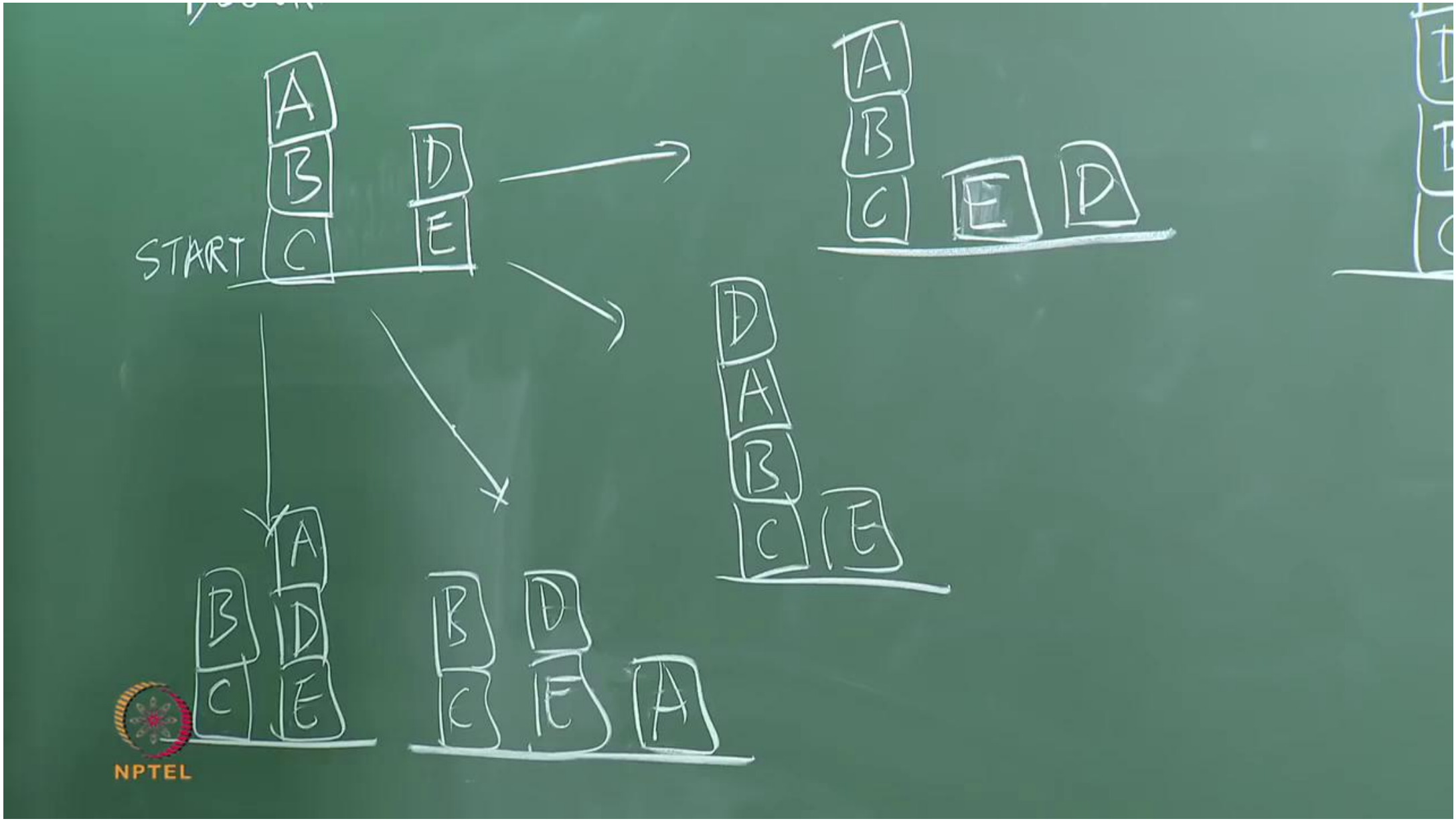

STEEPEST GRADIENT ASCE
NPTEL


BLOCKS WORLD DOMAIN
NPTEL


A
D
B
C
GOAL
E

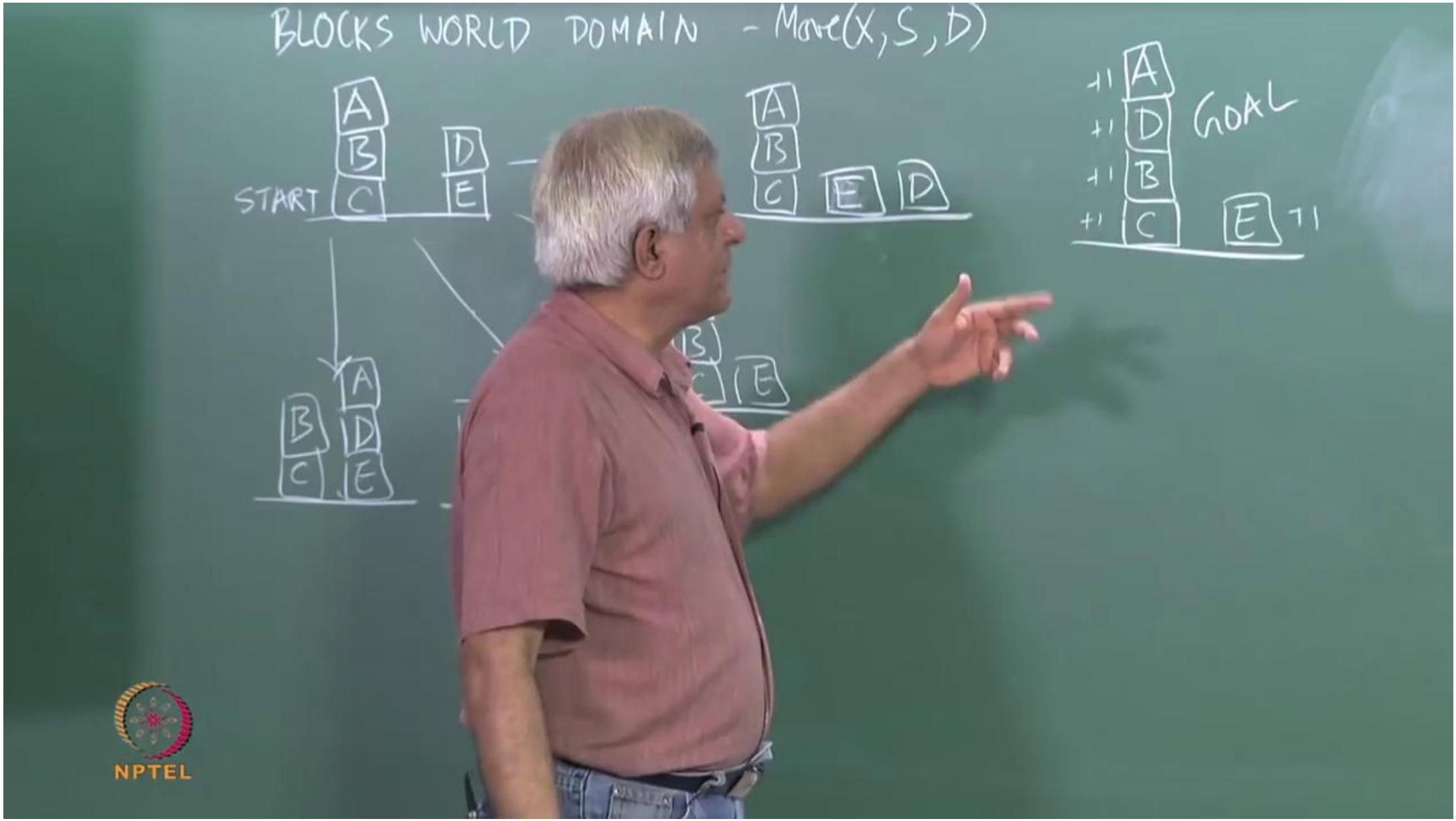

BLOCKS WORLD DOMAIN - Move(X,S,D)
START
A
B
C
A
D
B
C
GOAL
E
S
NPTEL

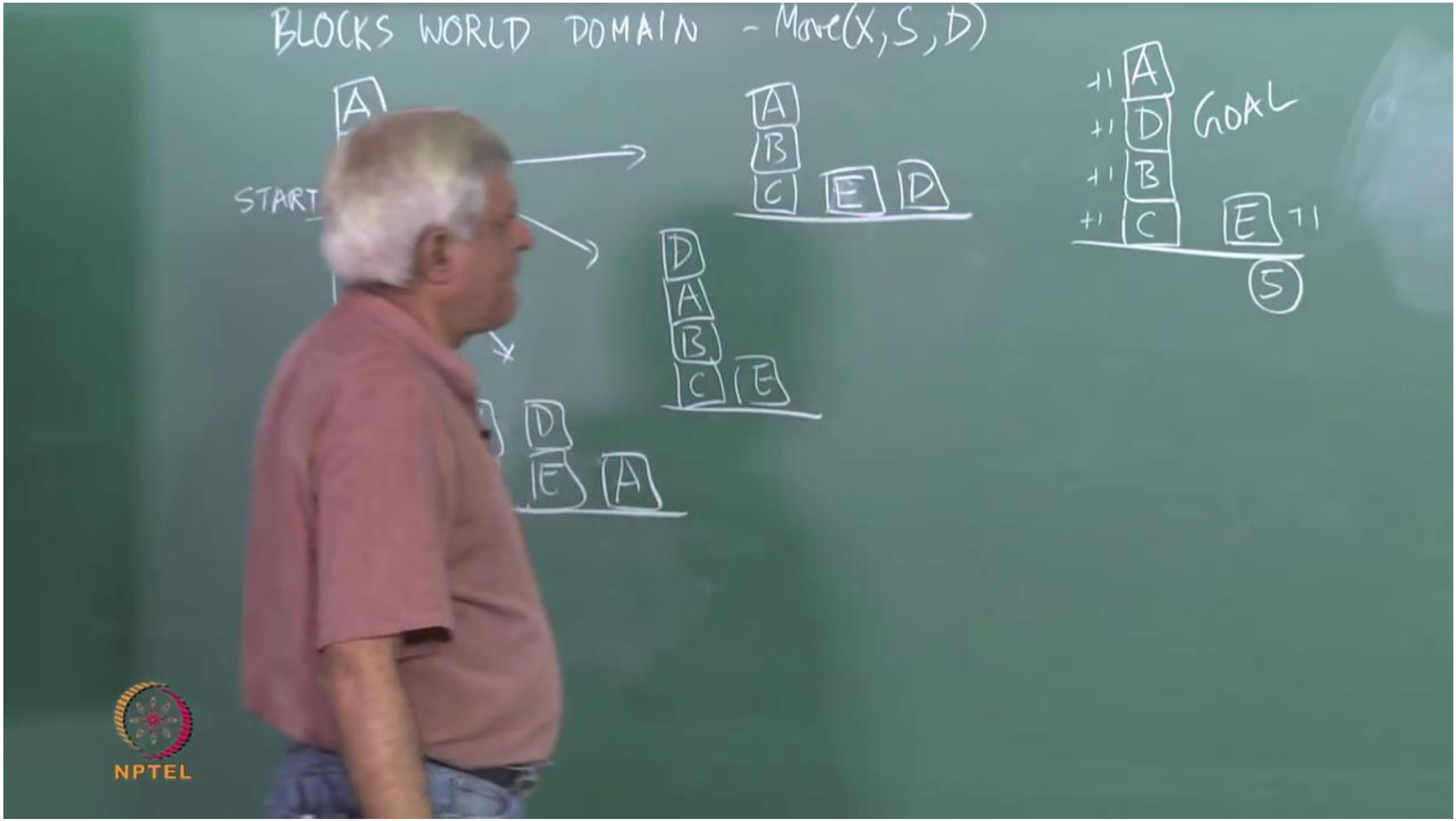

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
A
B
C
D
E
GOAL
A
D
B
C
E

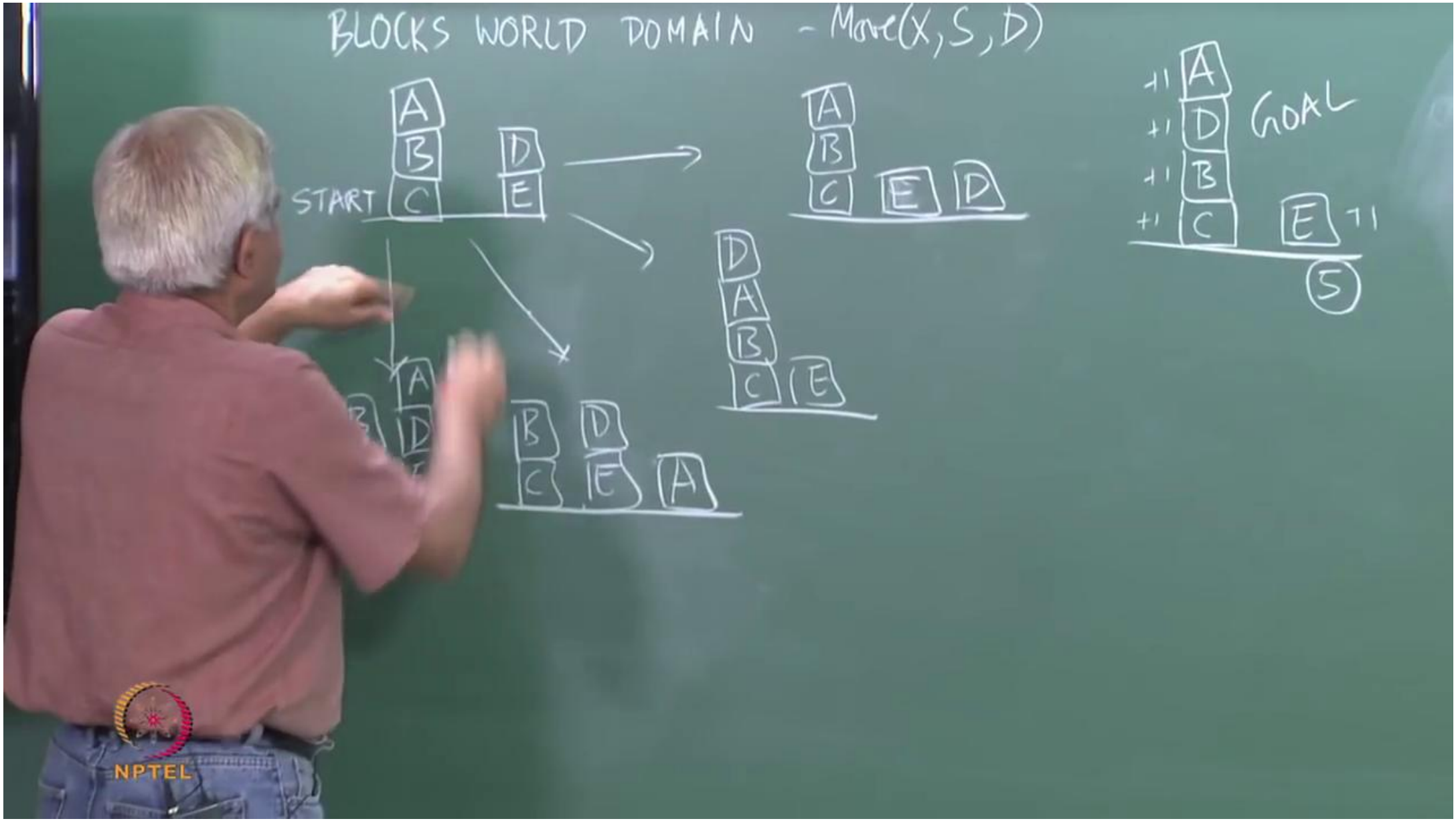

START
A
B
C
D
E
A
B
C
E
D
D
A
B
C
E
B
C
A
D
E
B
C
D
E
A

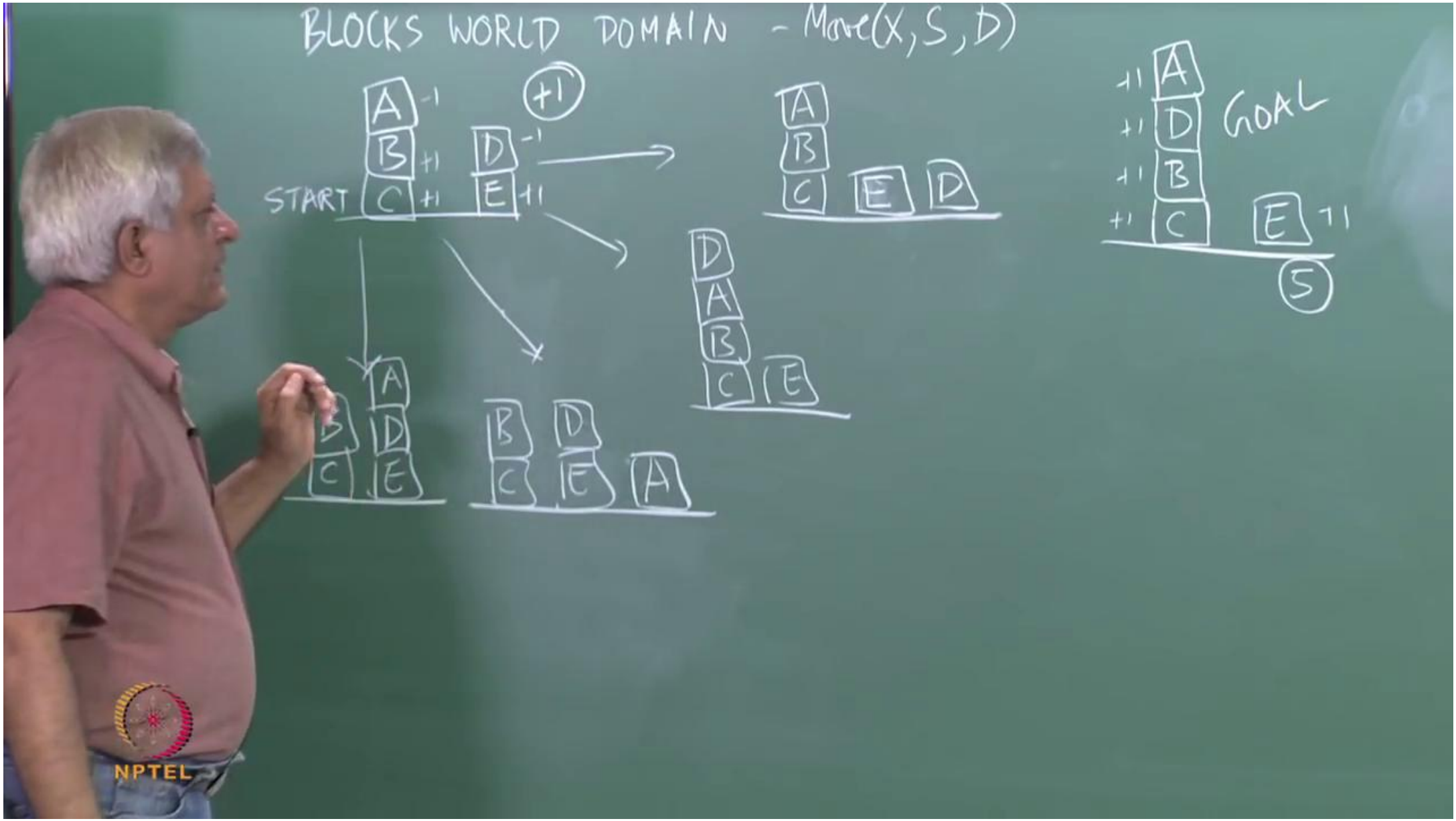
(5) ristic fn. $h(n)$

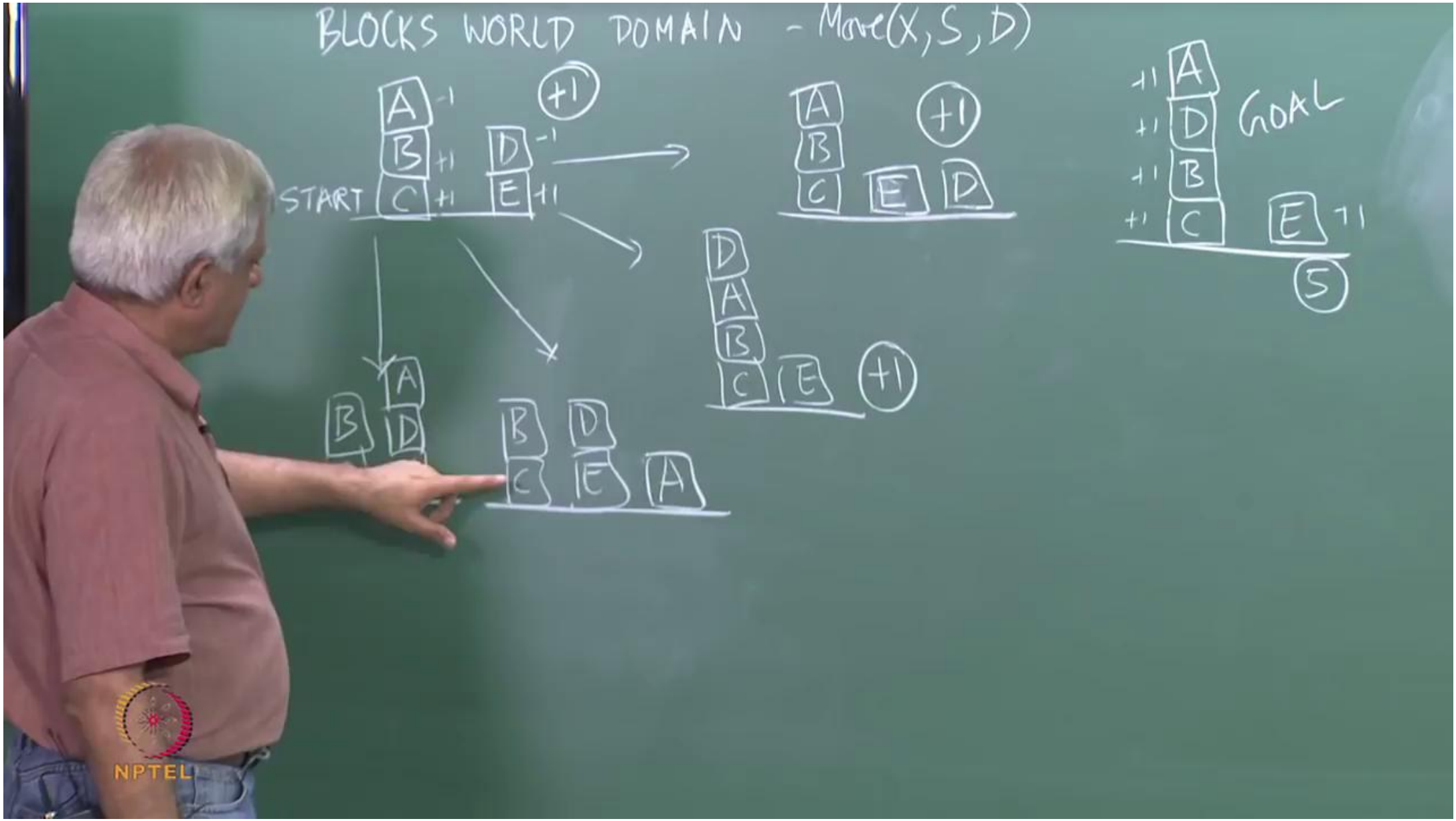
$h_1(n)$ = add 1 if block is on correct block / table
subtract 1 if it is on a wrong location

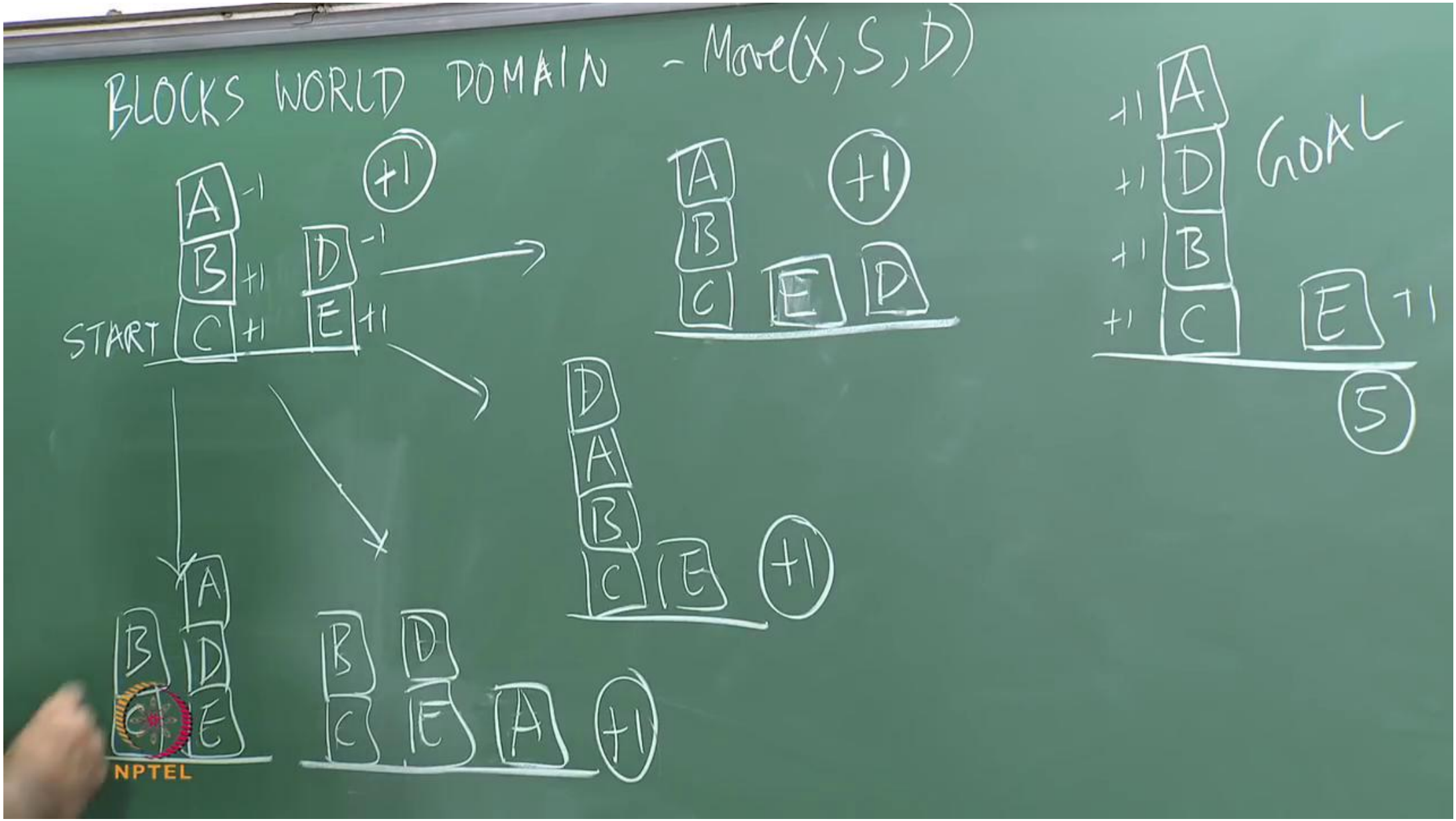

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
GOAL

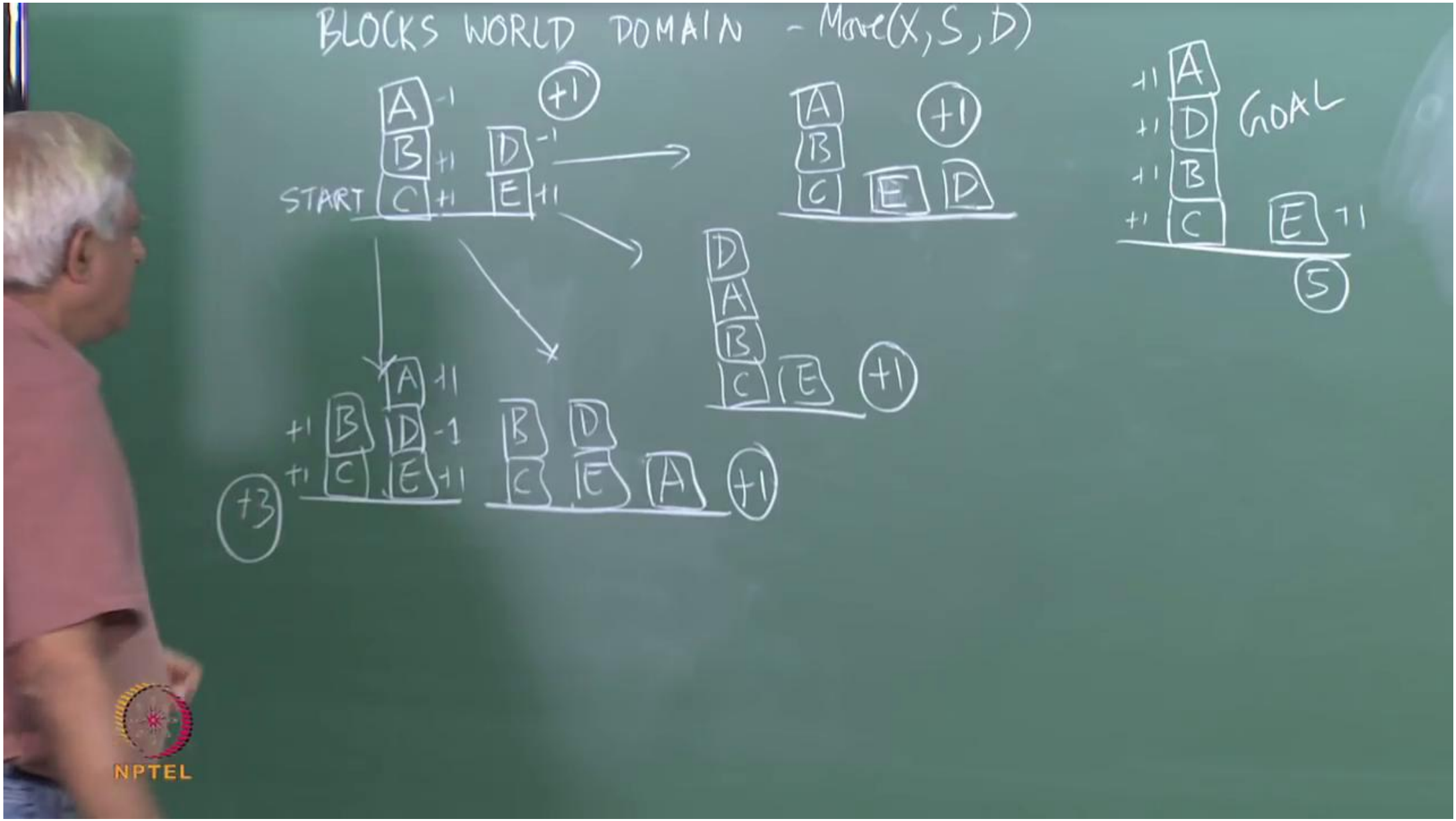

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
GOAL
NPTEL

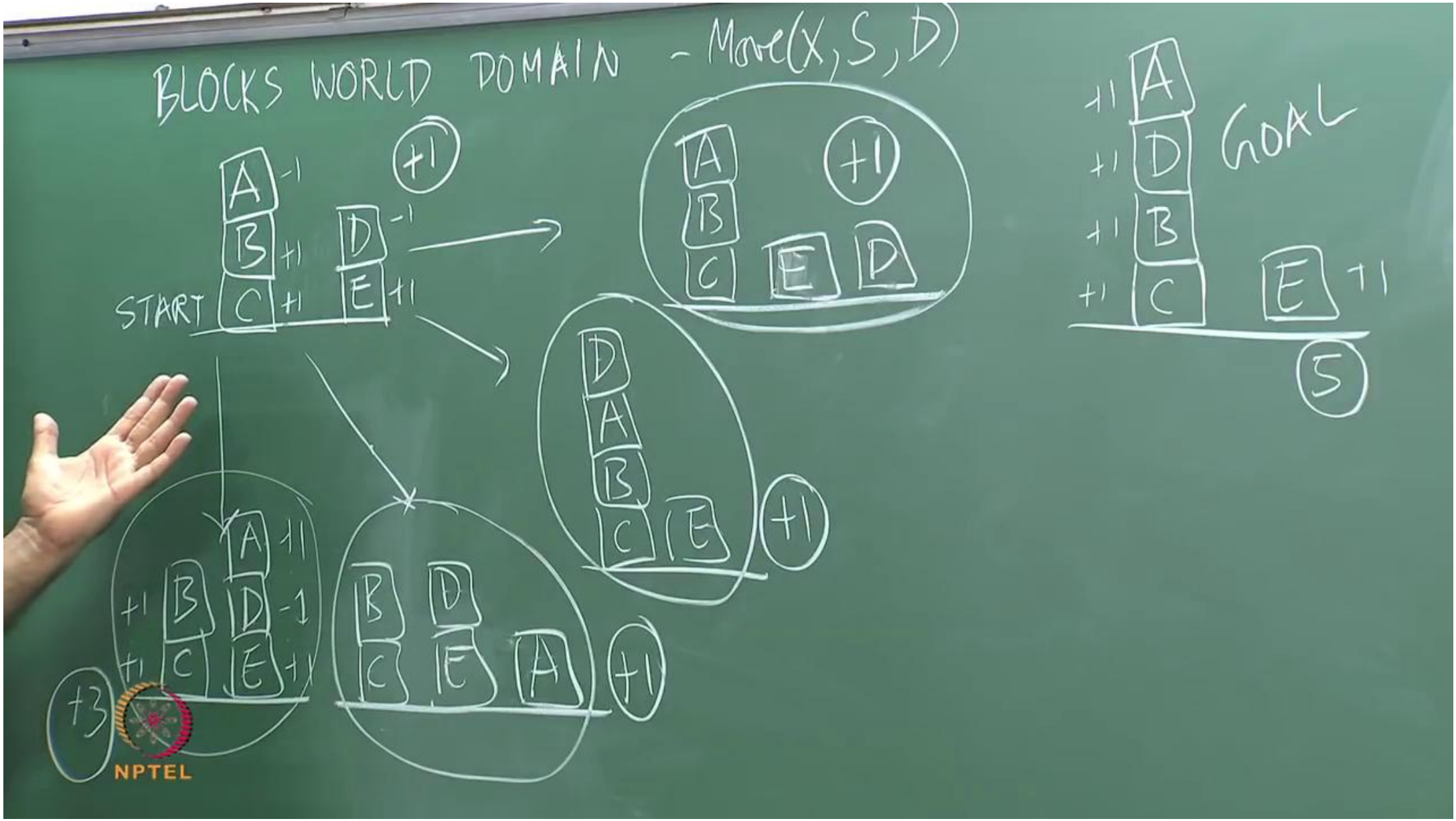

BLOCKS WORLD DOMAIN - Move(X,S,D)
START
GOAL
NPTEL

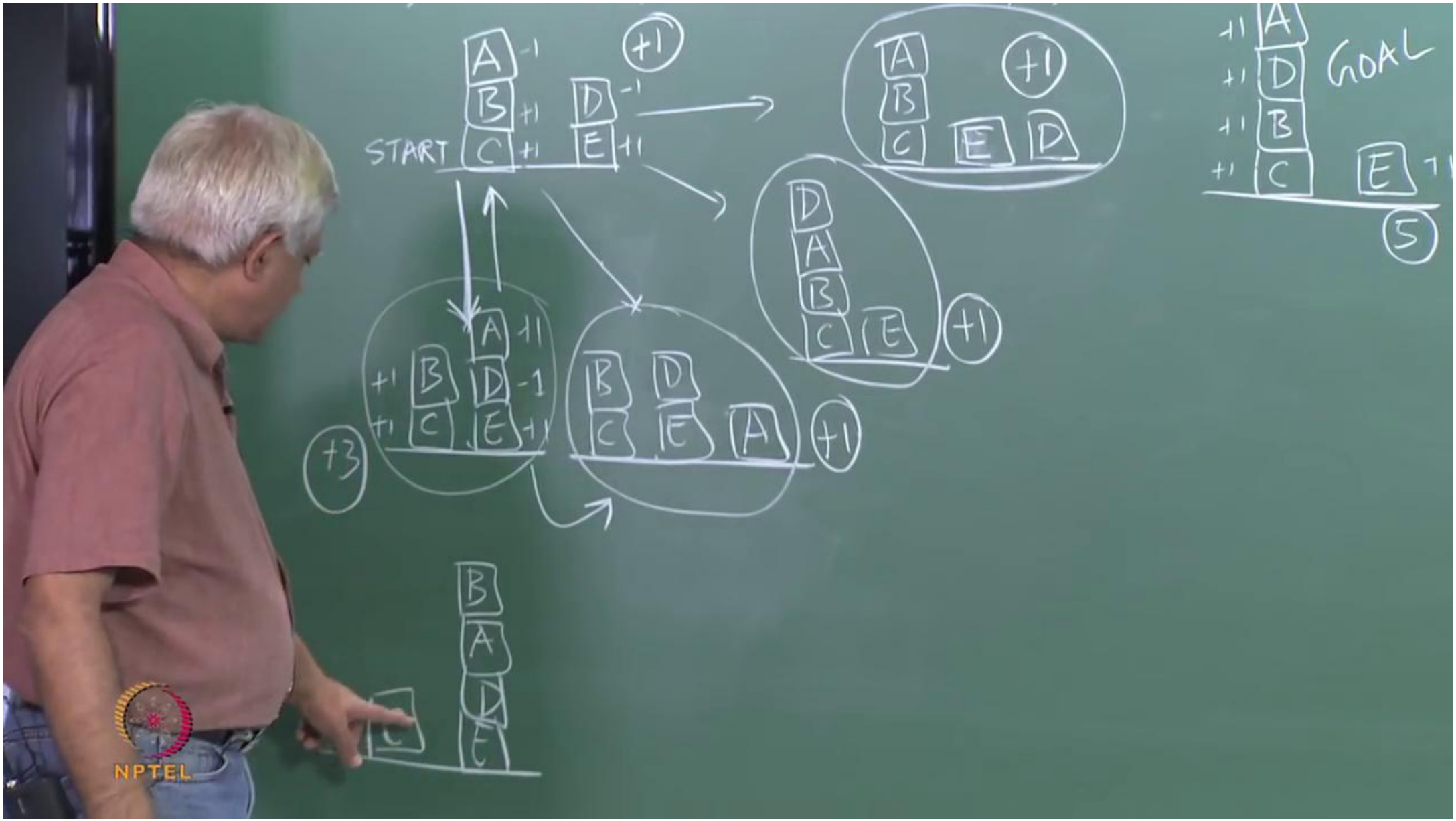

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
+1
GOAL
S

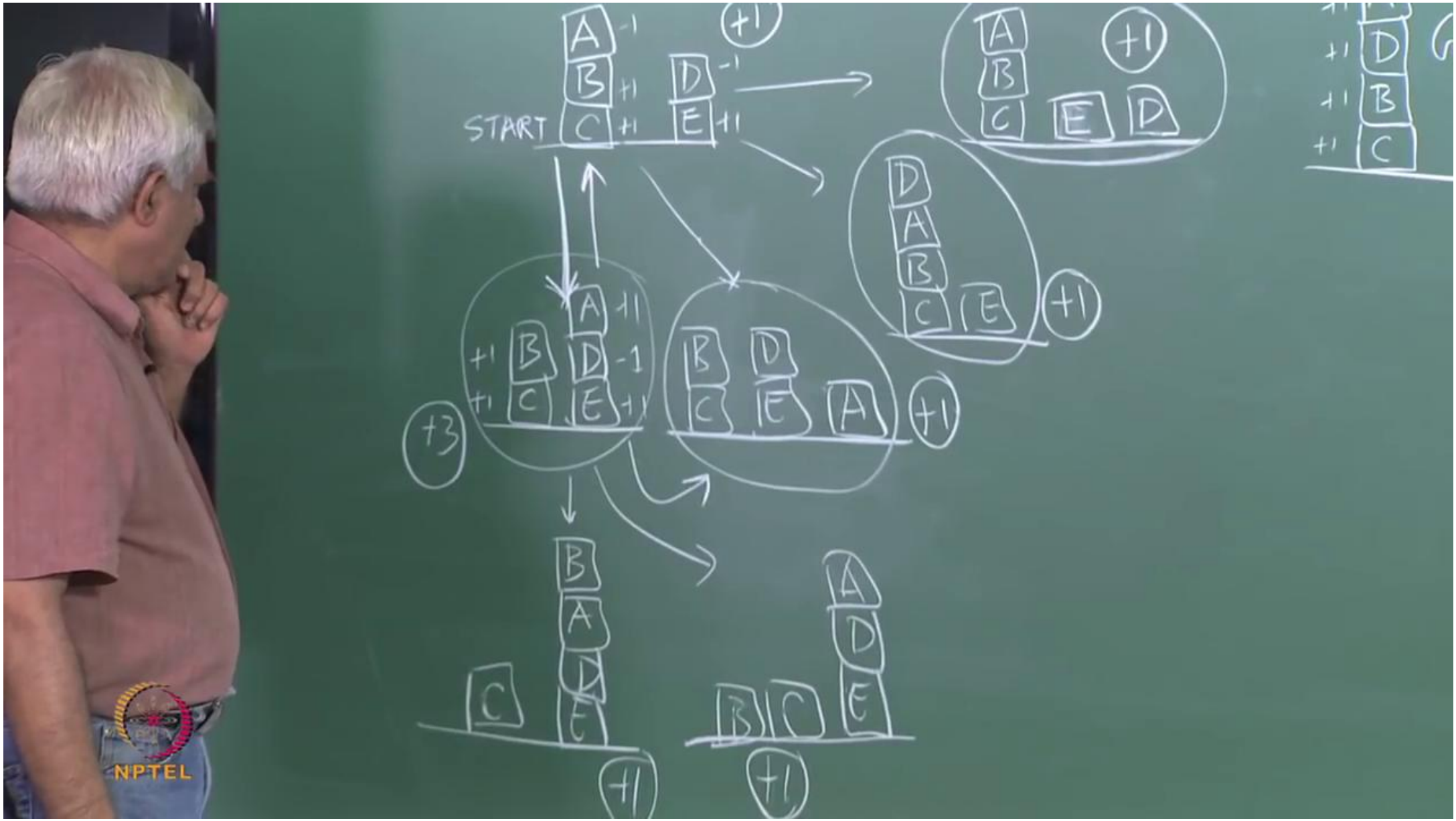

BLOCKS WORLD DOMAIN - Move(X,S,D)
START
GOAL
NPTEL

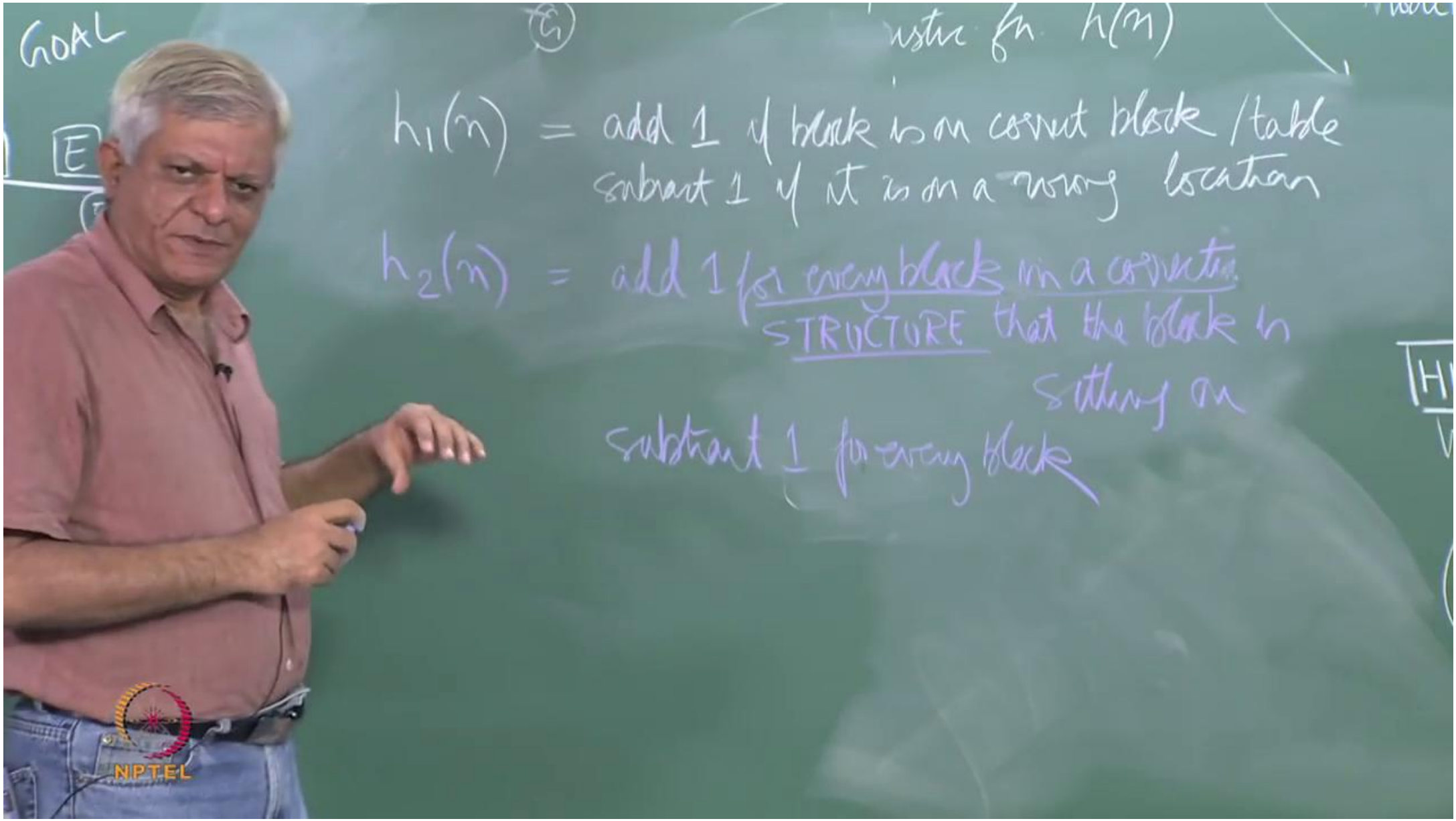

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
GOAL
NPTEL

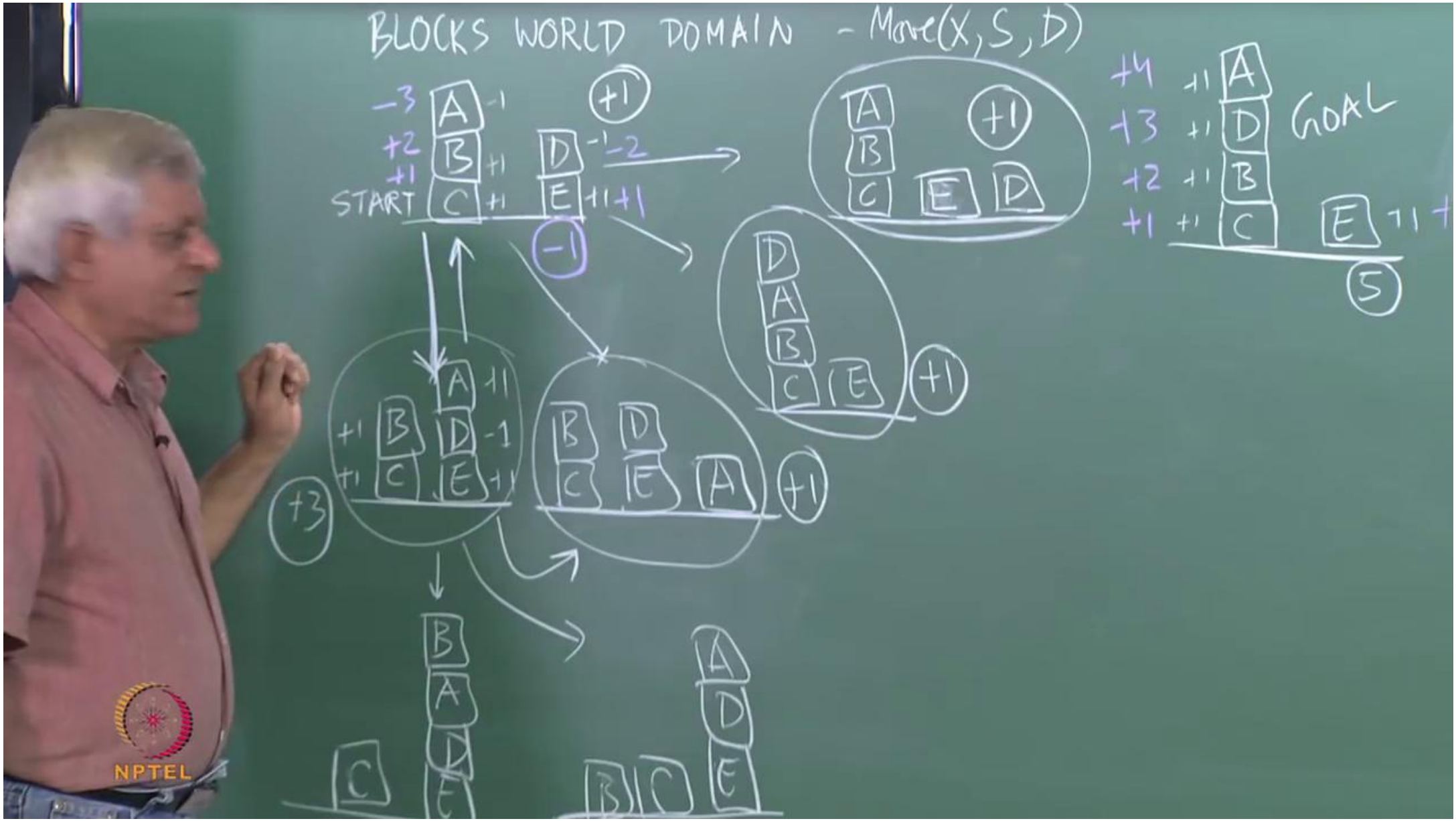

BLOCKS WORLD DOMAIN - Move(X,S,D)
START
GOAL
NPTEL

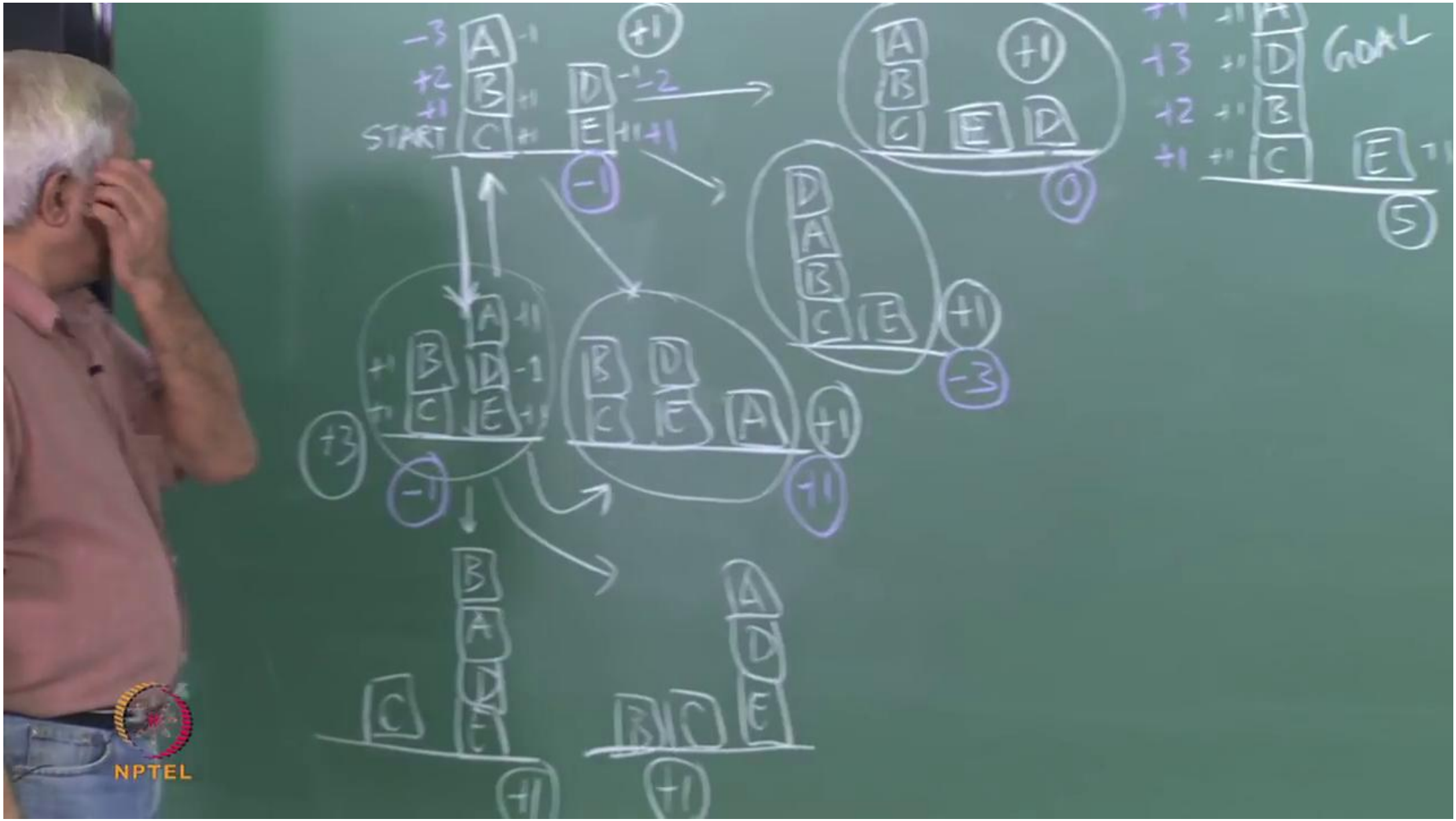

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
GOAL
NPTEL

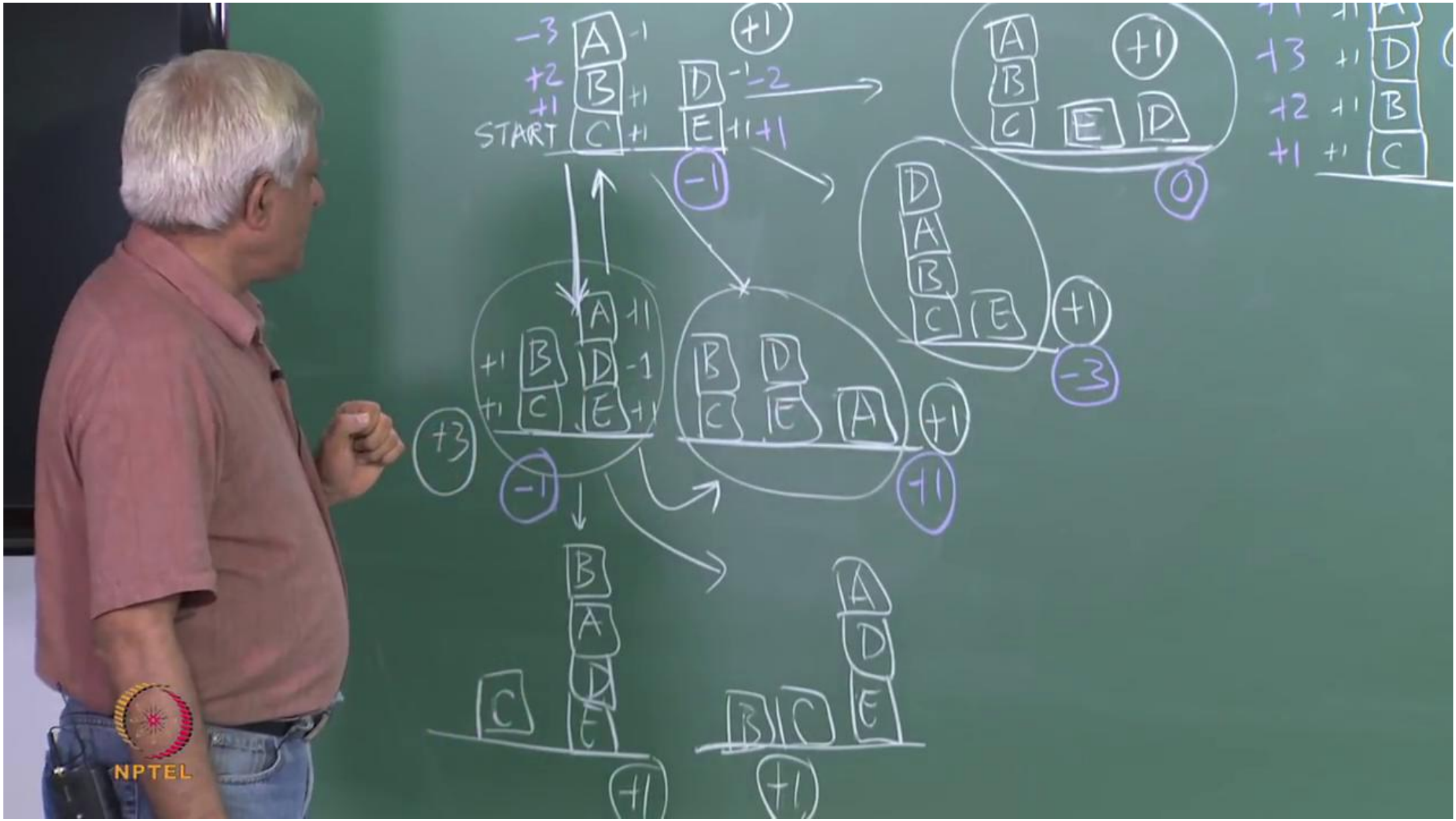

START
GOAL
+1
+3
5
NPTEL

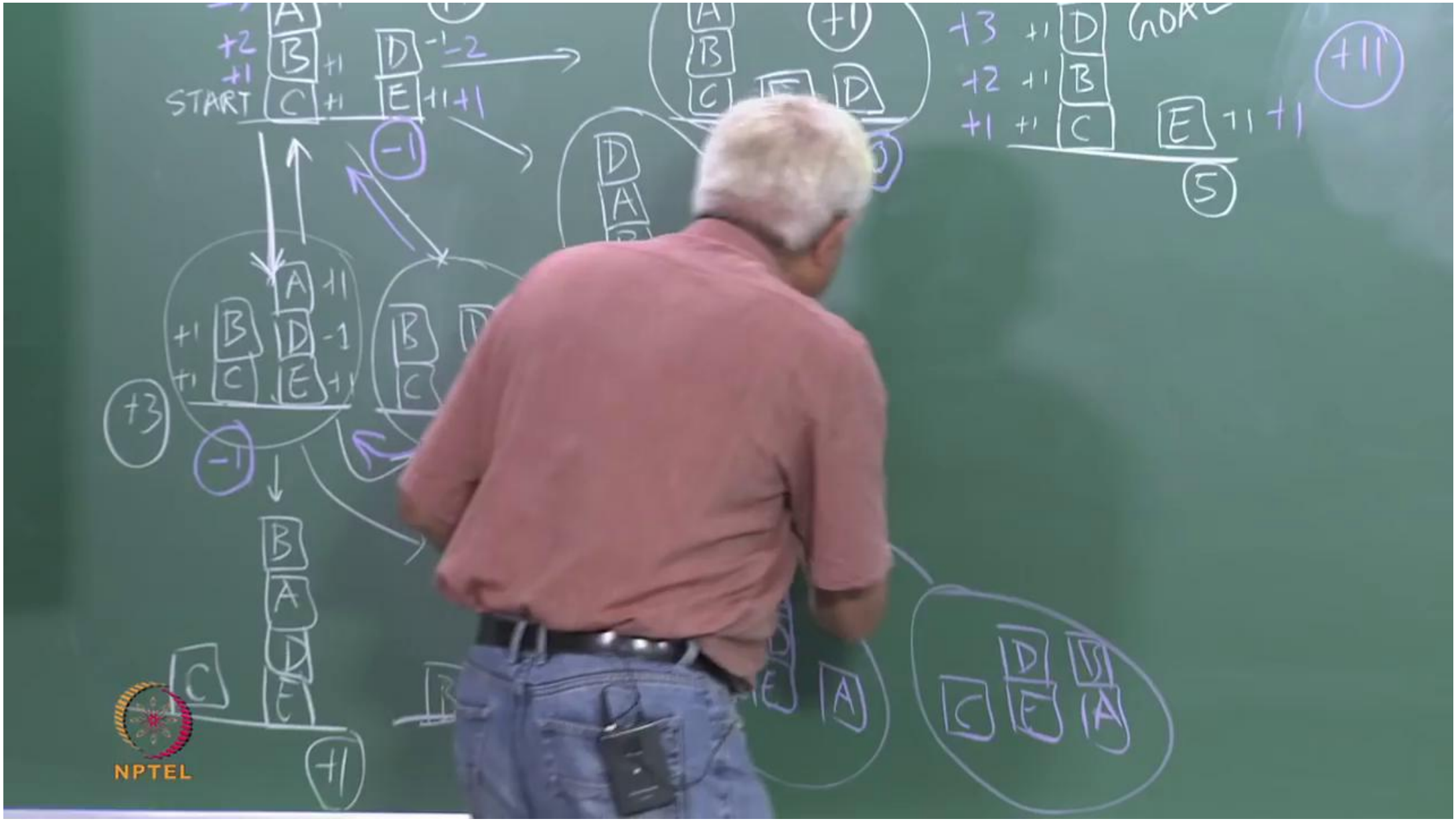

START
A
B
C
D
E
-1
+1
+3
NPTEL

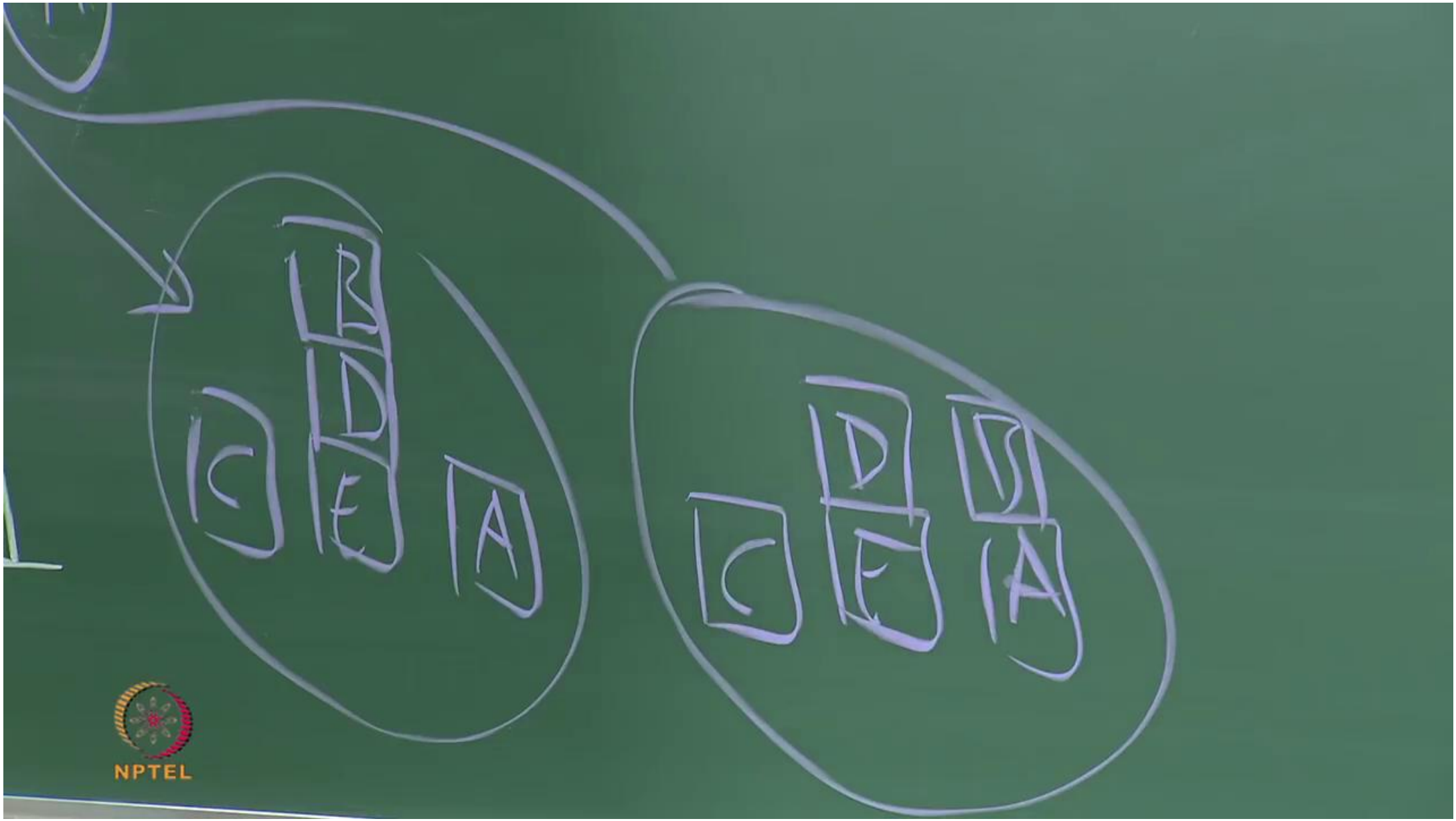

GOAL
$h_1(n)$ = add 1 if block is on correct block / table
subtract 1 if it is on a wrong location
$h_2(n)$ = add 1 for every block in a correct STRUCTURE that the block is sitting on
subtract 1 for every block

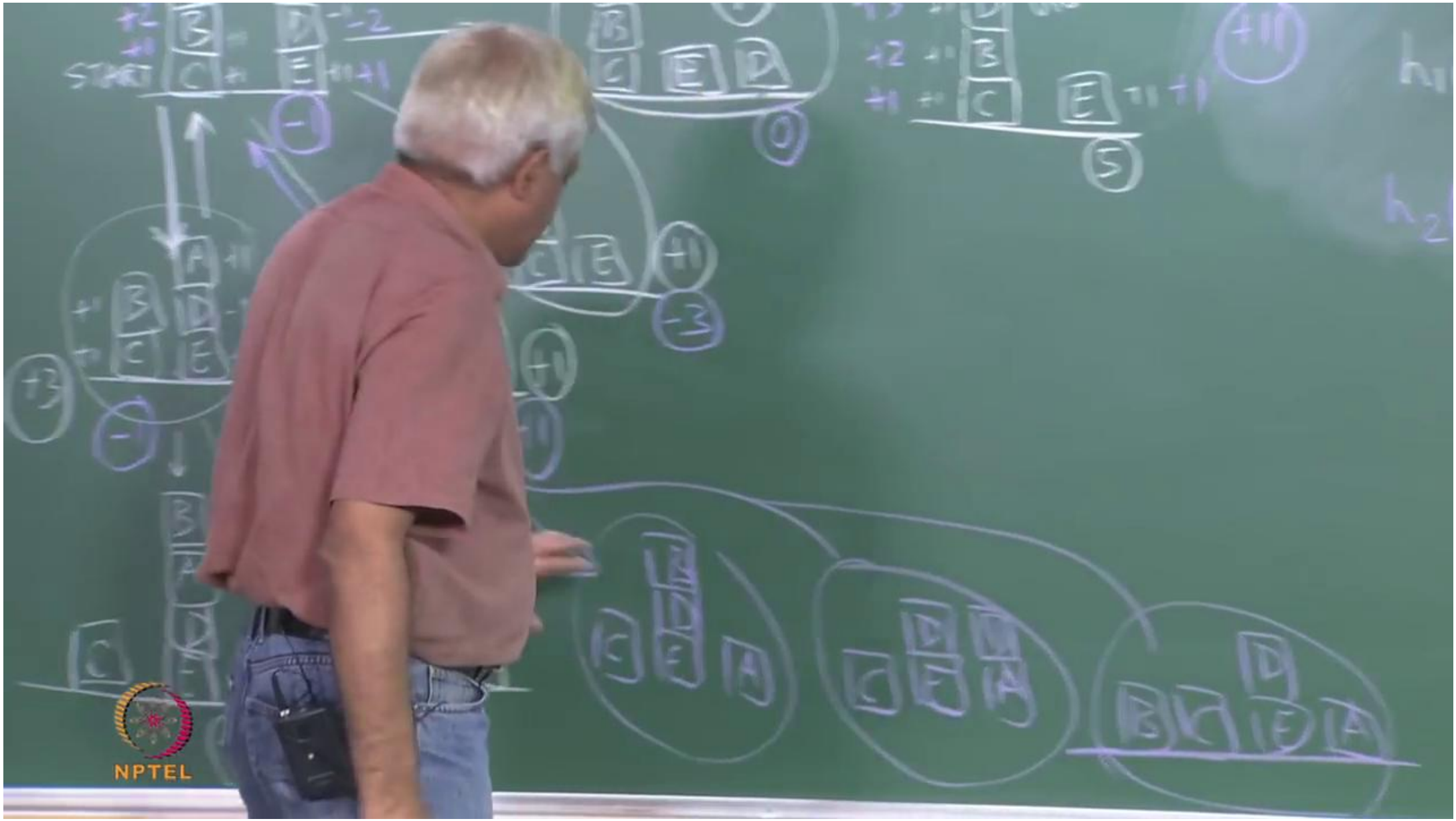

BLOCKS WORLD DOMAIN - Move(X, S, D)
START
GOAL
NPTEL

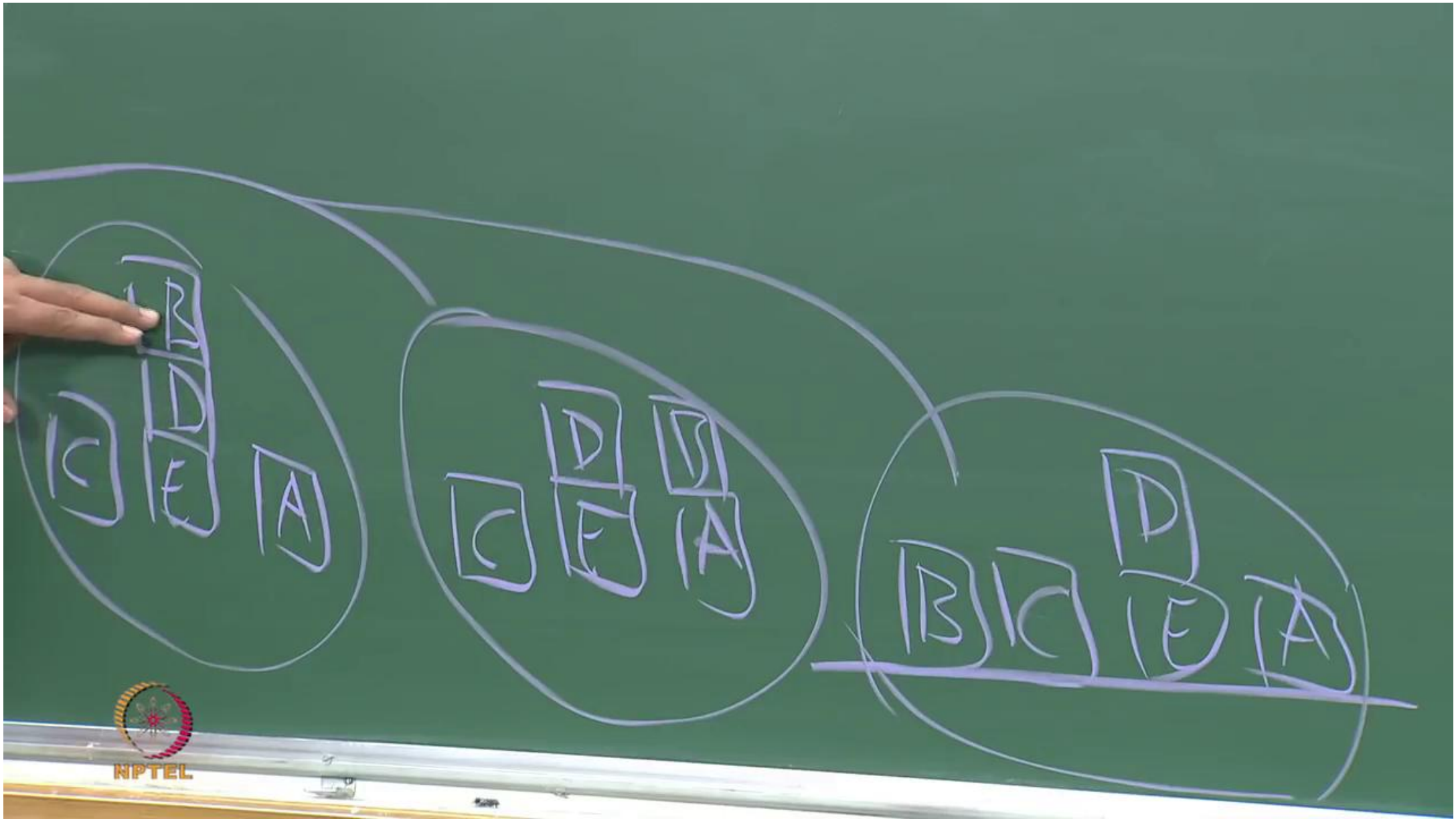

START
GOAL
NPTEL

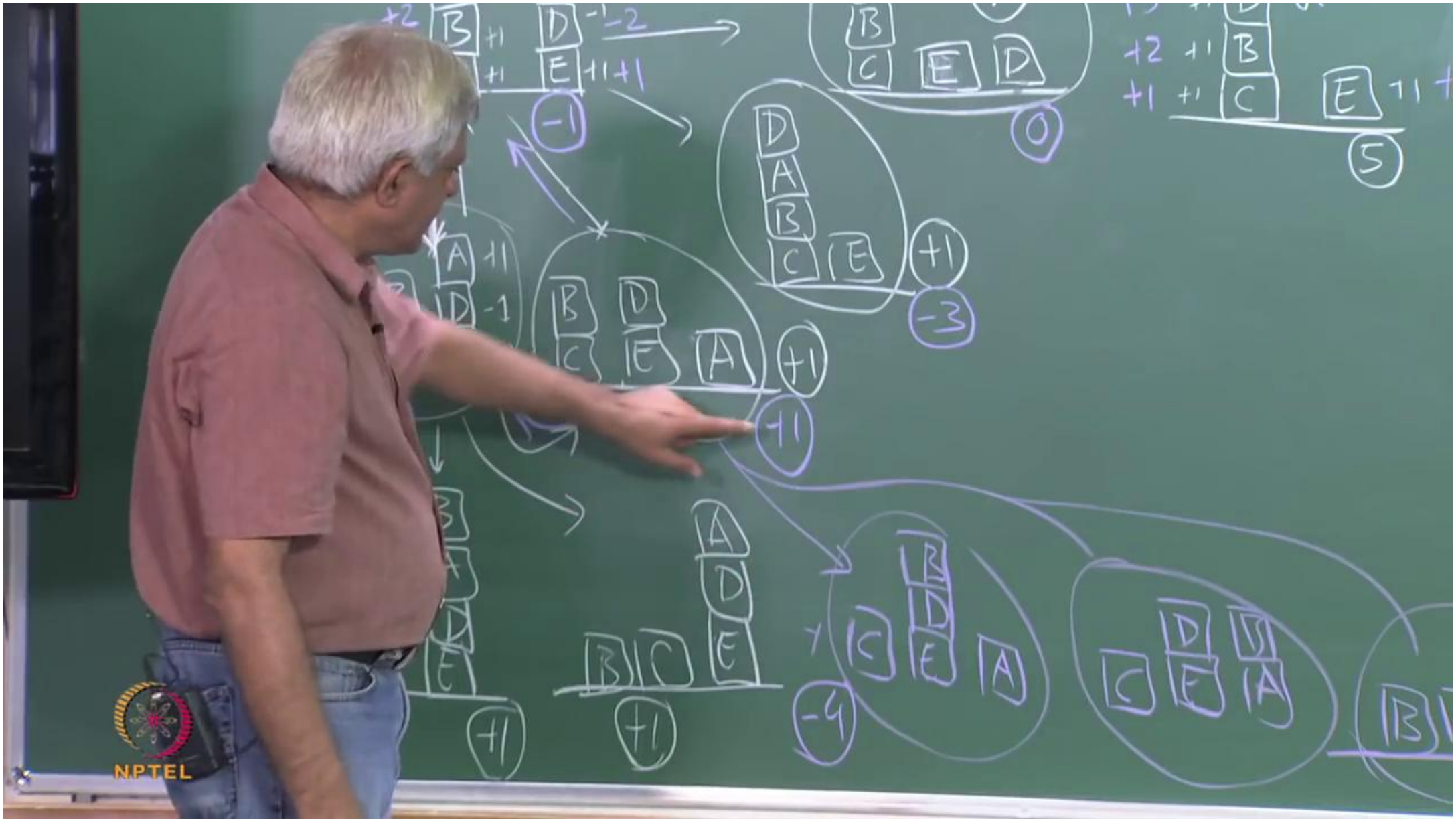

START
A
B
C
D
E
NPTEL

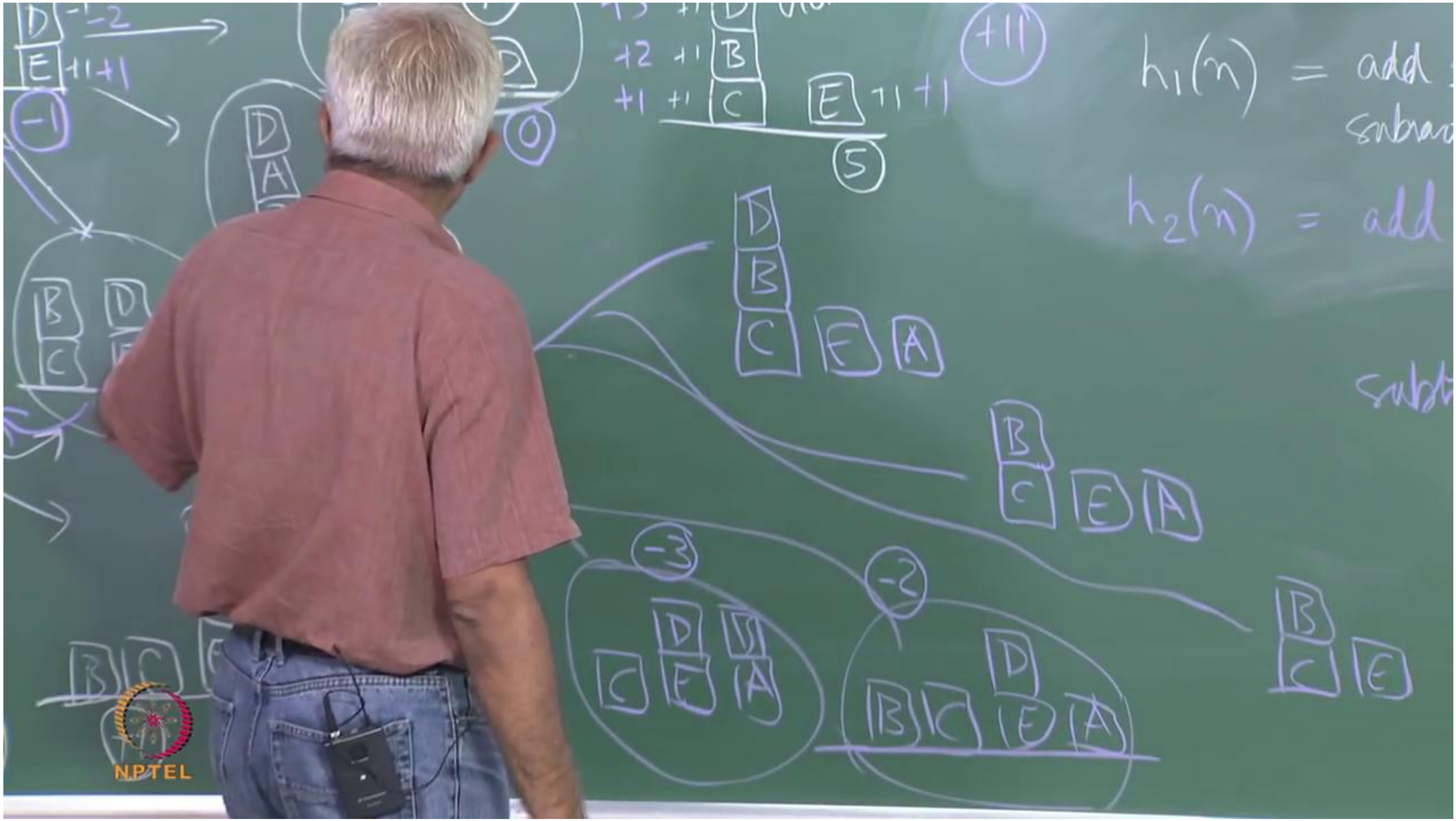

START
+2
+1
-1
-2
+3
+11
-1
+3
-1
+11
NPTEL

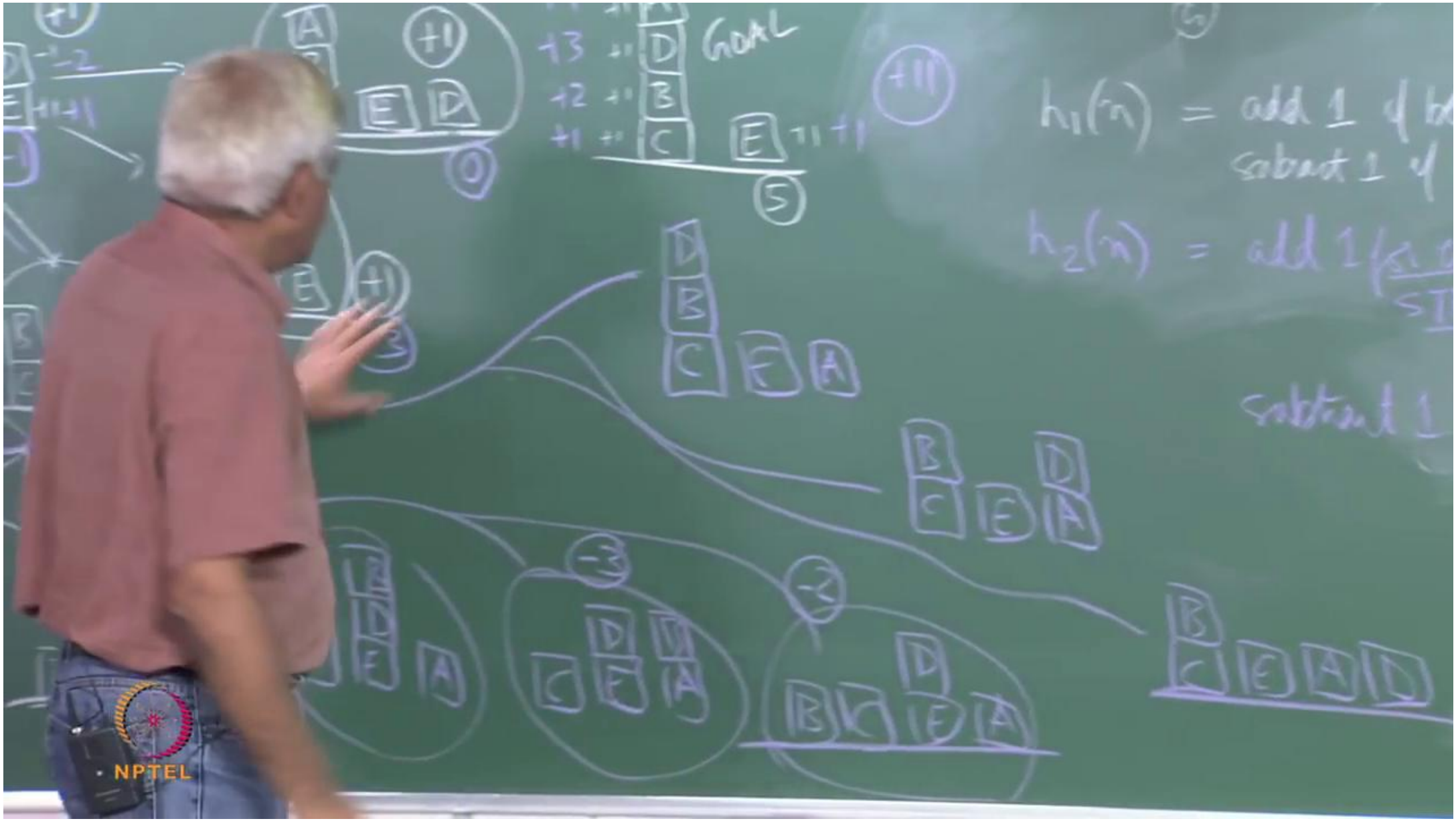

C
D
E
A
C
D
E
A

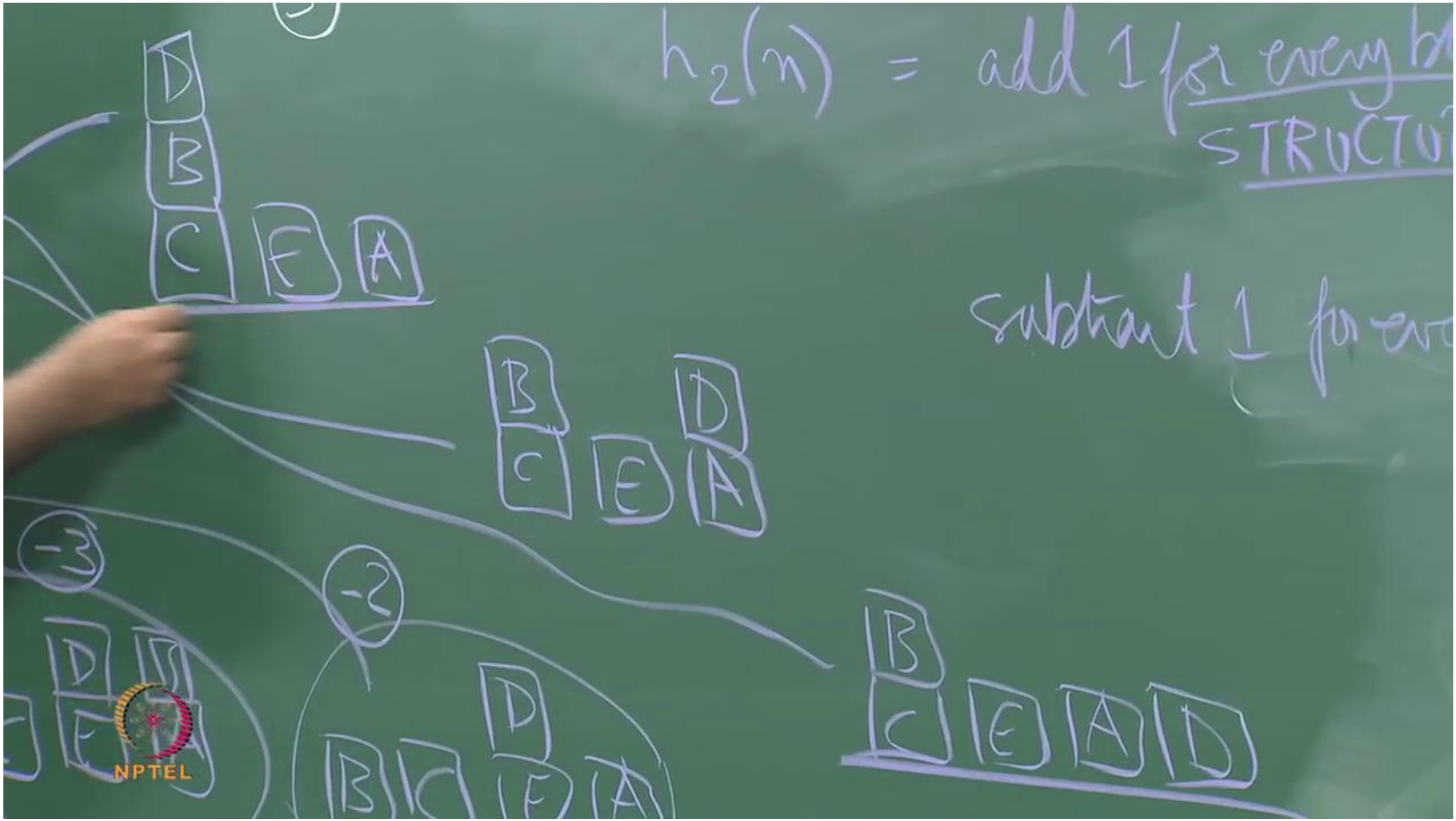

NPTEL

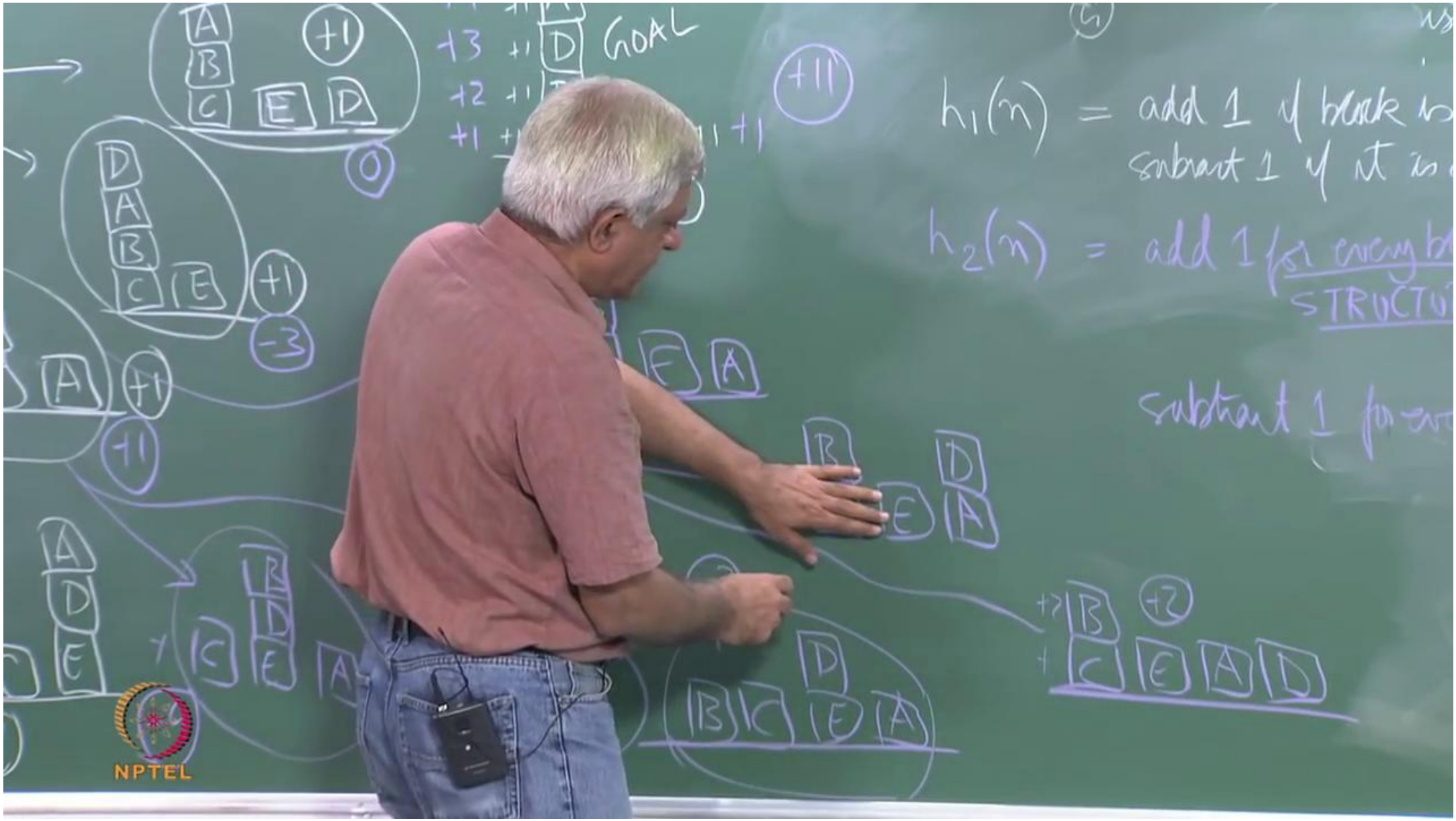

B
D
C
E
A
D
D
C
E
A
D
B
C
E
A

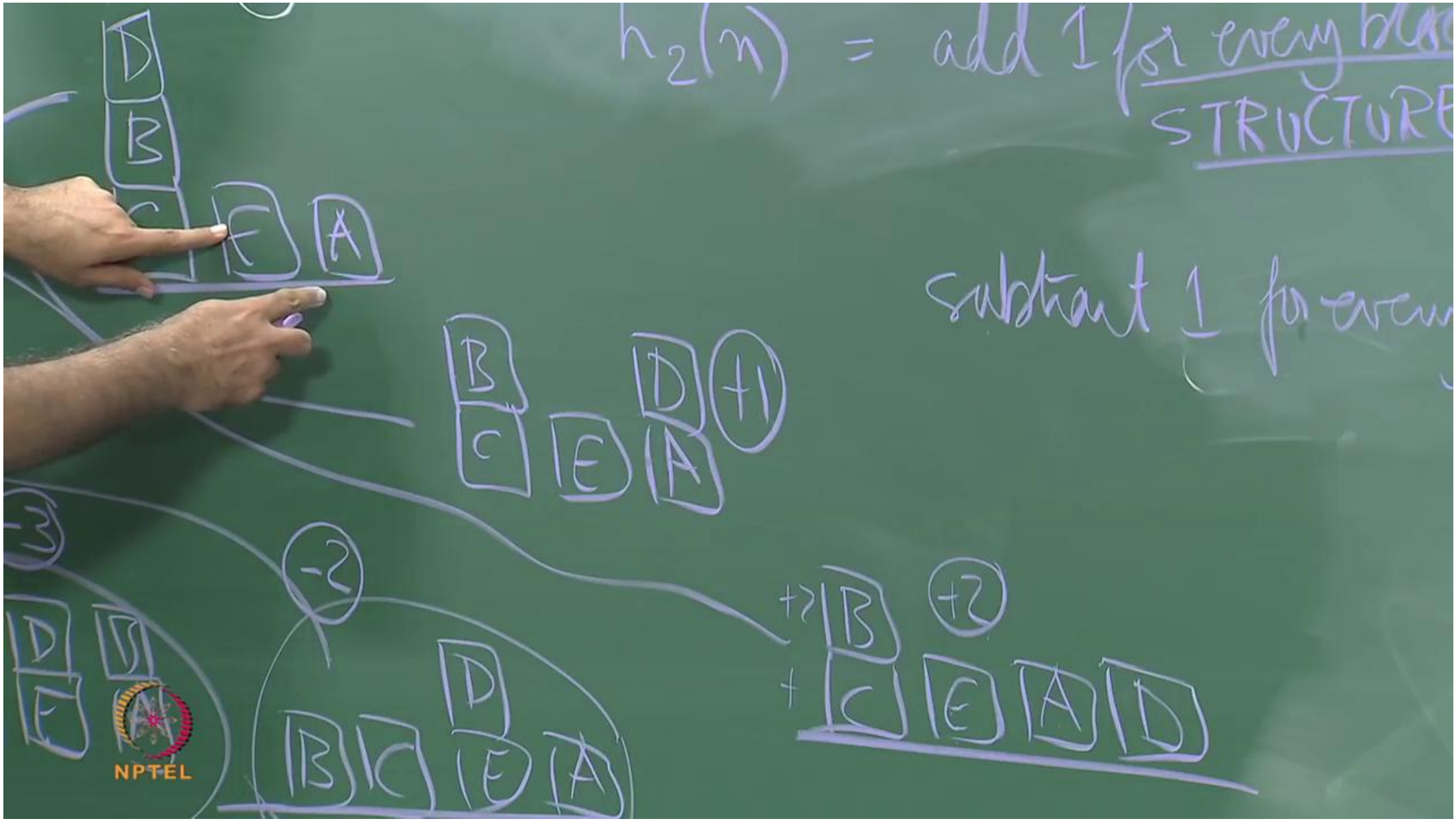

NPTEL

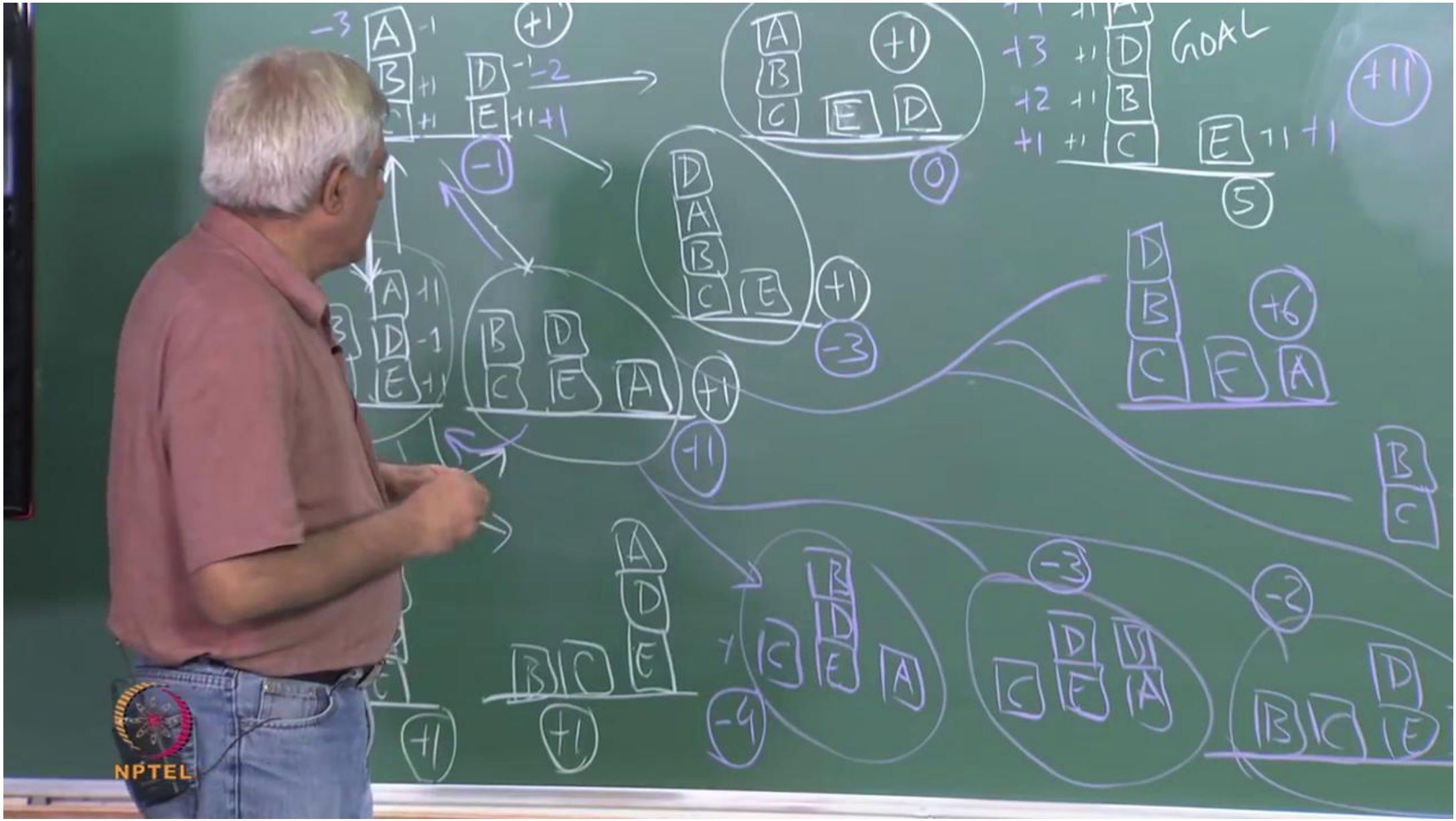

h1(n) = add
h2(n) = add
NPTEL

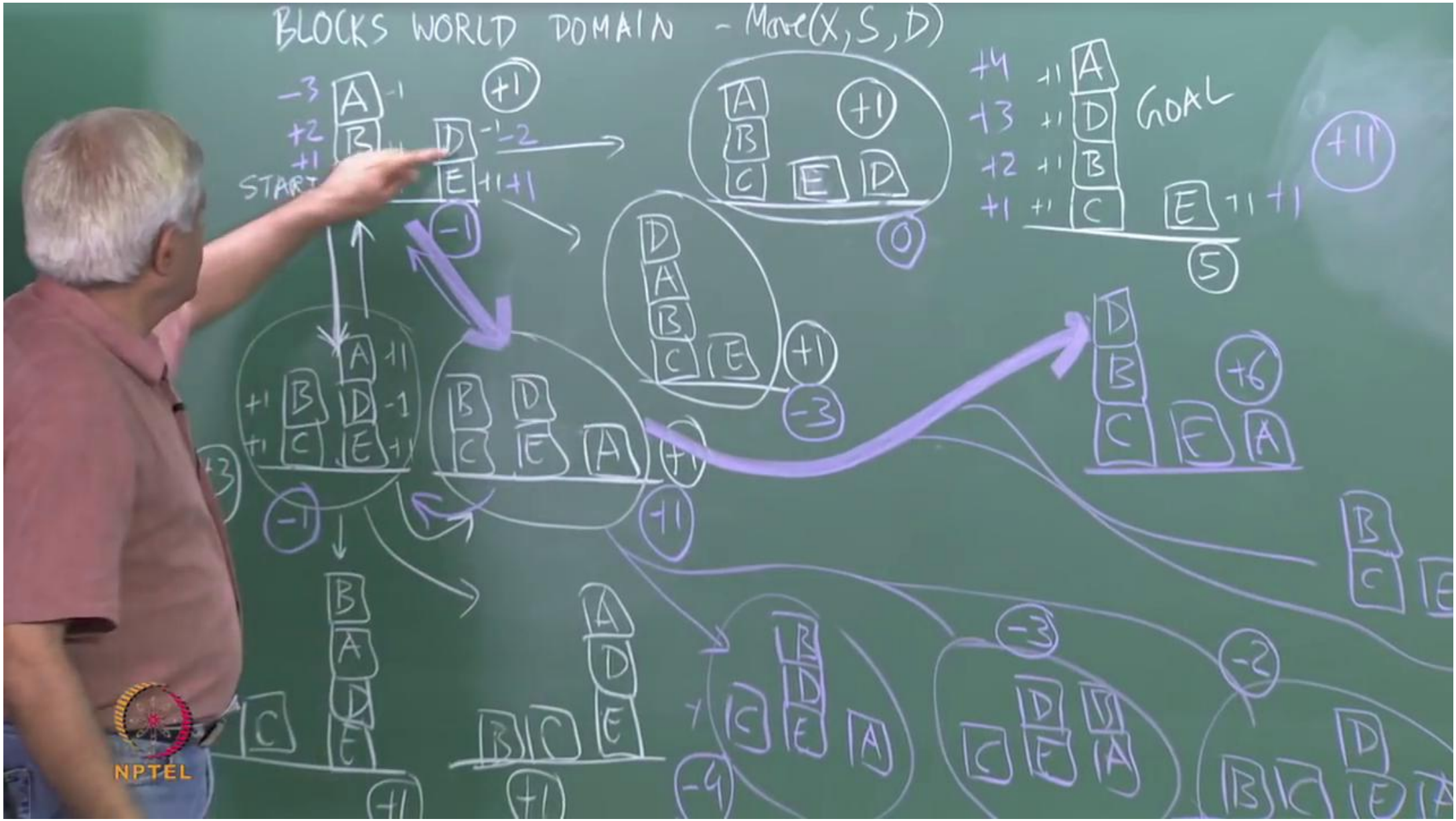

GOAL
$h_1(n)$ = add 1 if
Subtract 1 if
$h_2(n)$ = add 1
Subtract 1

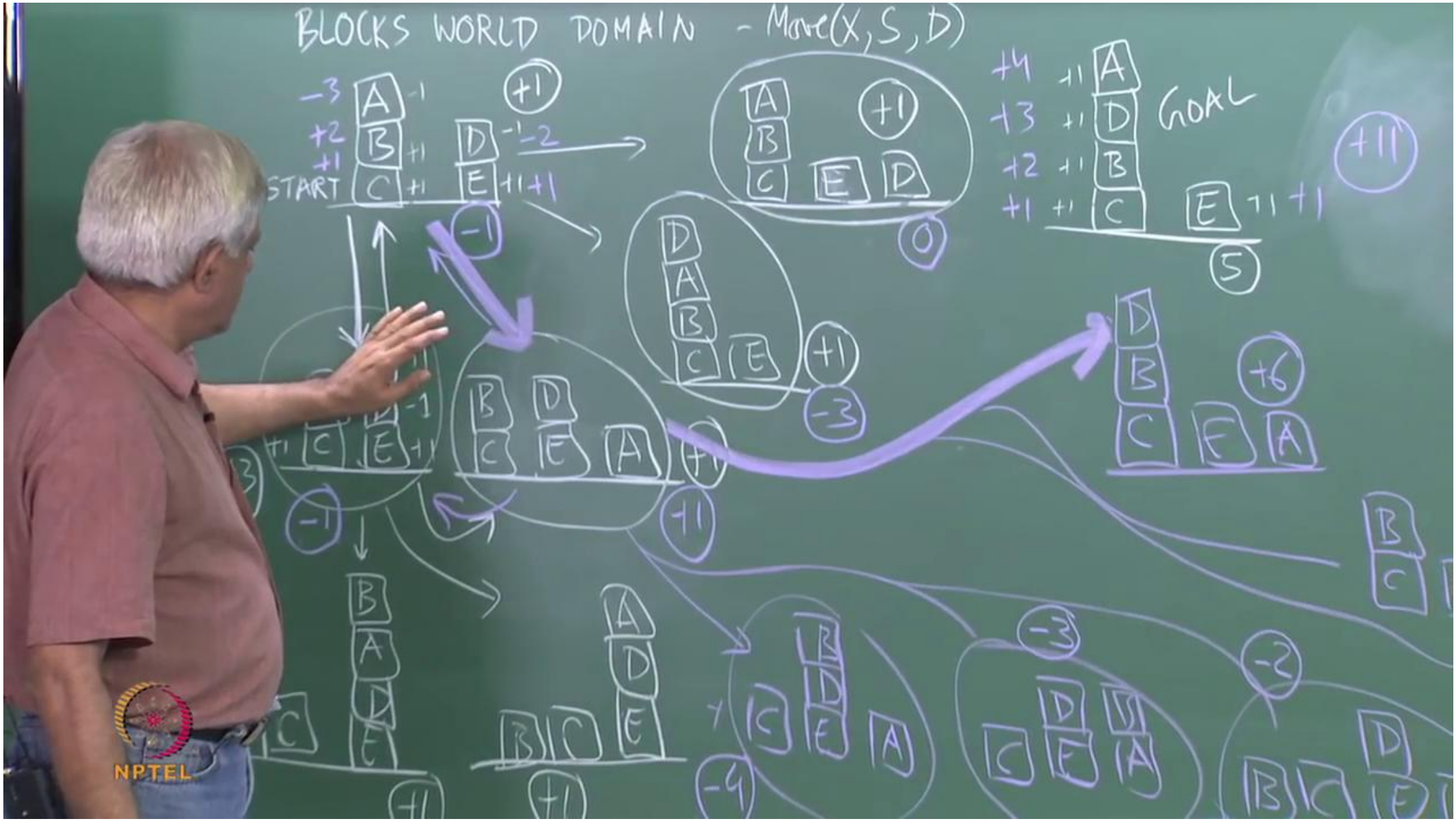

h2(n) = add 1 for every
STRUCTU
subtract 1 for ev
D
B
C
F
A
B
C
E
D
A
-3
-2
B
C
E
A
D
NPTEL

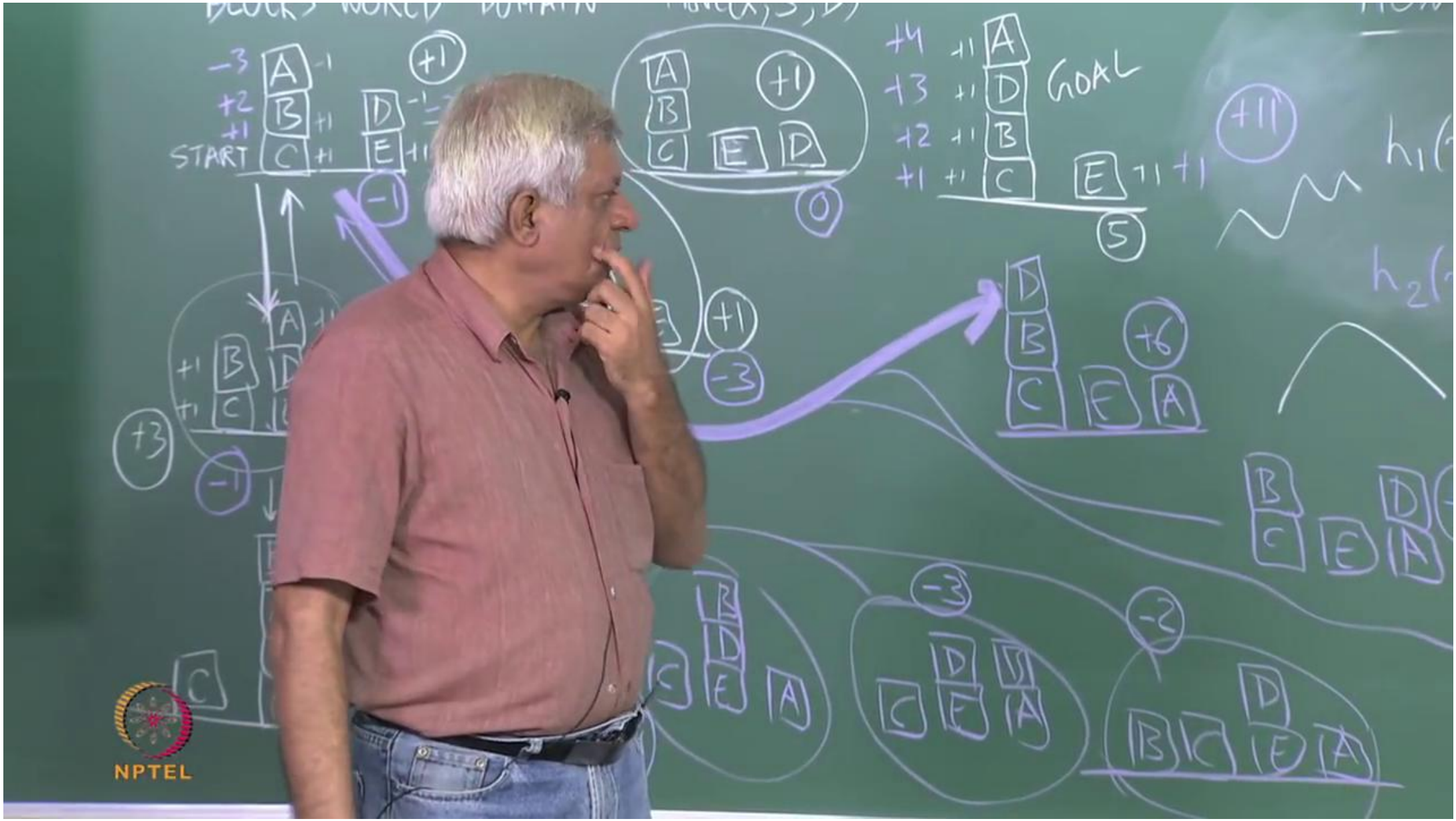

GOAL
+1
+3
+2
+1
+11
-3
0
$h_1(n)$ = add 1 if block is
subtract 1 if it is
$h_2(n)$ = add 1 for
STRUCTU
subtract 1 for
+2
NPTEL


h2(n) = add 1 for every
STRUCTURE
subtract 1 for every
+1
-2
+2
NPTEL


GOAL
NPTEL


BLOCKS WORLD DOMAIN - Move(X,S,D)
START
GOAL
NPTEL


BLOCKS WORLD DOMAIN - Move(X,S,D)
START
GOAL
NPTEL


START
GOAL
NPTEL