Week 3 notes prepared by Baskaran Sankaranarayanan.

Tie Breaking a Pool of Best Nodes

The MoveGen determines the order in which neighbours are generated. The algorithms determine the order in which nodes are inspected. Greedy algorithms select the best node in each iteration, and often there will be a pool of best nodes with the same (best) cost and we do not know which ones lead to a solution. In real world problems, an additional criteria (external to the algorithm) can be applied to filter the pool of best nodes, or else we can randomly choose a node from the pool. Each choice may potentially lead to a different solution or no solution at all.

Therefore, a pool of best nodes makes it difficult to discuss/grade assignments and exam papers. To reduce the difficulty, we use label-order (alphabetical order of node labels) to select nodes from the pool of best nodes.

When multiple nodes have the same (best) cost, sort those nodes by node label (ascending) and select from the head of the sorted list.

This general purpose tie-breaking rule helps us to focus on the algorithm and not on the finer details of a (made up) toy problem.

As an example, the SAT problem from week 3 is reworked using the tie-breaking rule.

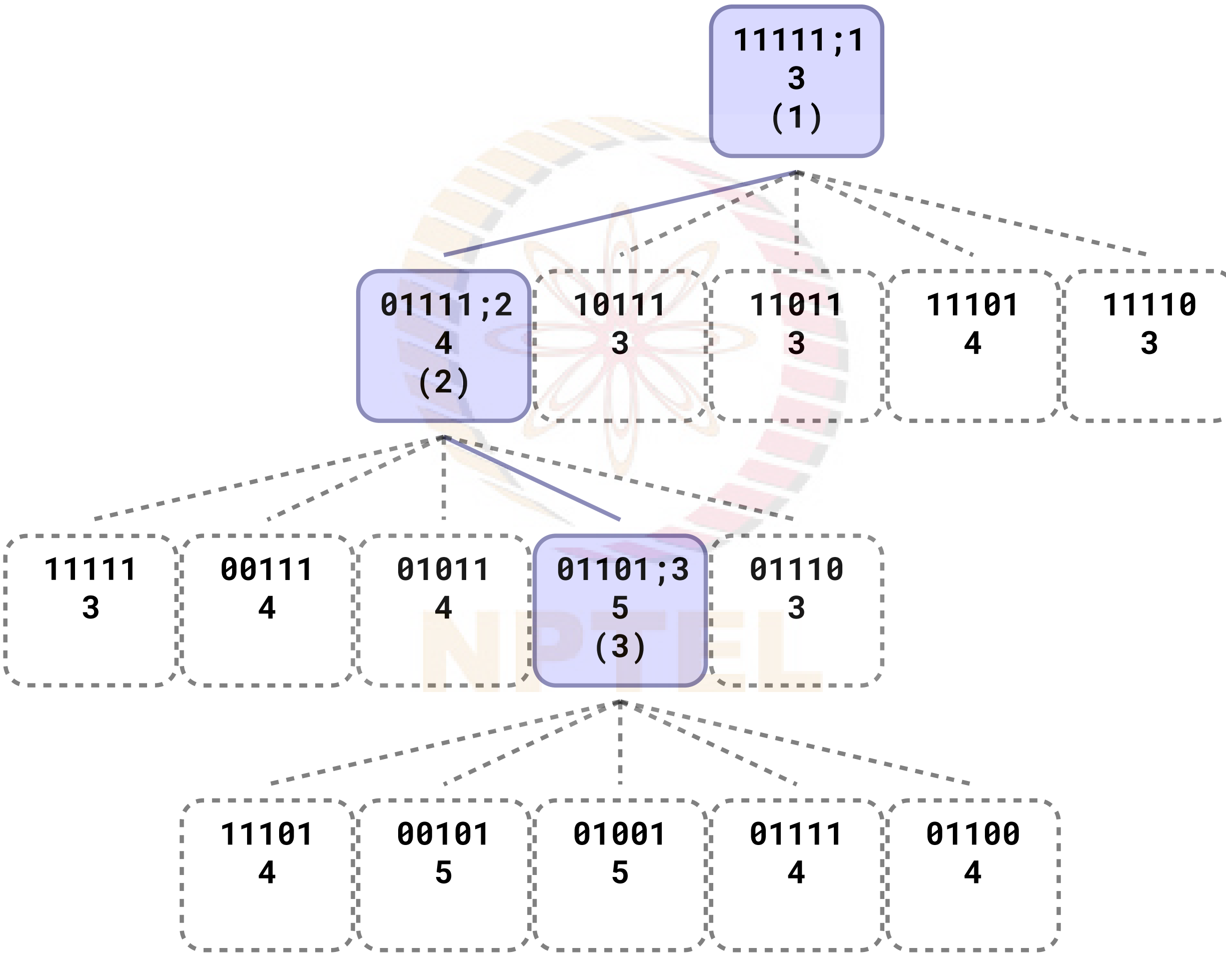
The second column in the table below shows the node labels in alphabetical order (which in this case is also the numerical order).

Formula Evaluation

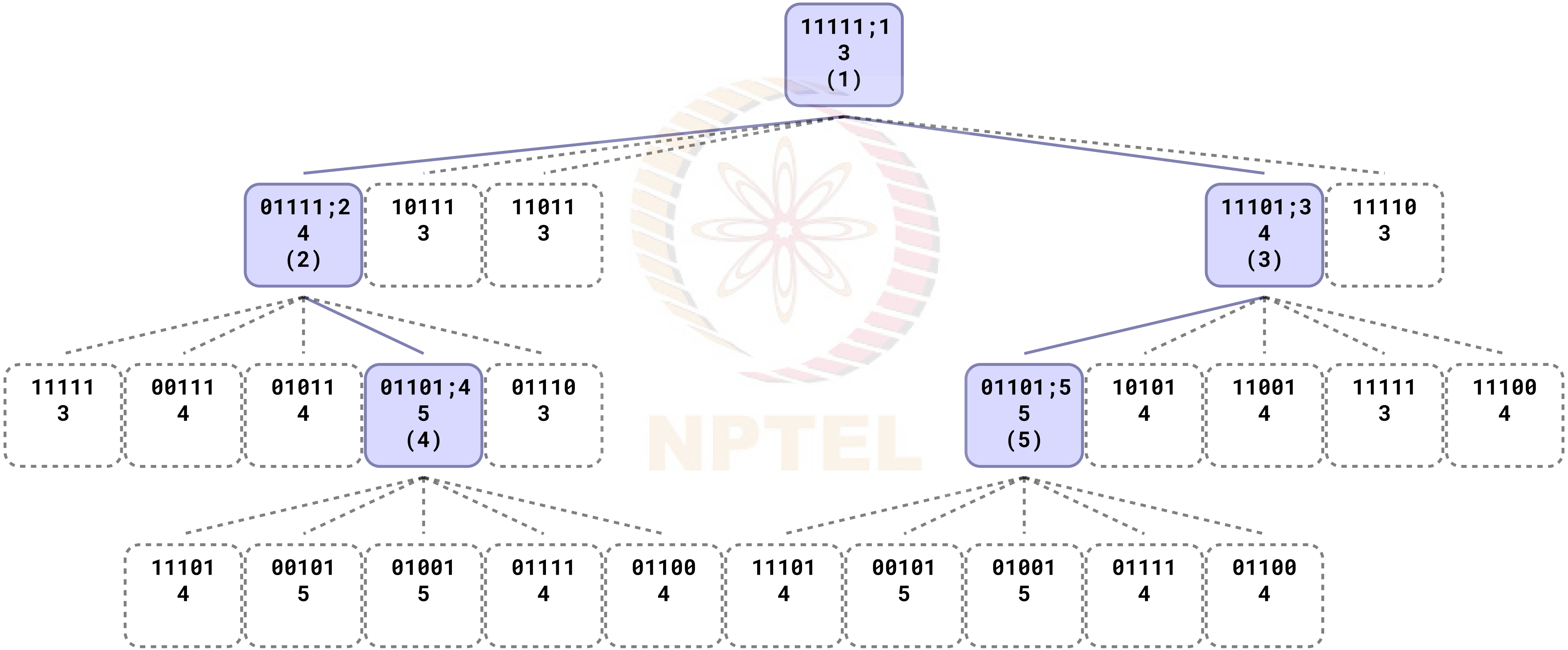
$F = (b \lor \neg c) \land (c \lor \neg d) \land (\neg b) \land (\neg a \lor \neg e) \land (e \lor \neg c) \land (\neg c \lor \neg d)$

S.No.	abcde	h(F)	eval(F)	$(b \lor \neg c)$	$(c \lor \neg d)$	$(\neg b)$	$(\neg a \lor \neg e)$	$(e \lor \neg c)$	$(\neg c \lor \neg d)$
1	00000	6	1	1	1	1	1	1	1
2	00001	6	1	1	1	1	1	1	1
3	00010	5	0	1	0	1	1	1	1
4	00011	5	0	1	0	1	1	1	1
5	00100	4	0	0	1	1	1	0	1
6	00101	5	0	0	1	1	1	1	1
7	00110	3	0	0	1	1	1	0	0
8	00111	4	0	0	1	1	1	1	0
9	01000	5	0	1	1	0	1	1	1
10	01001	5	0	1	1	0	1	1	1
11	01010	4	0	1	0	0	1	1	1
12	01011	4	0	1	0	0	1	1	1
13	01100	4	0	1	1	0	1	0	1
14	01101	5	0	1	1	0	1	1	1
15	01110	3	0	1	1	0	1	0	0
16	01111	4	0	1	1	0	1	1	0
17	10000	6	1	1	1	1	1	1	1
18	10001	5	0	1	1	1	0	1	1
19	10010	5	0	1	0	1	1	1	1
20	10011	4	0	1	0	1	0	1	1
21	10100	4	0	0	1	1	1	0	1
22	10101	4	0	0	1	1	0	1	1
23	10110	3	0	0	1	1	1	0	0
24	10111	3	0	0	1	1	0	1	0
25	11000	5	0	1	1	0	1	1	1
26	11001	4	0	1	1	0	0	1	1
27	11010	4	0	1	0	0	1	1	1
28	11011	3	0	1	0	0	0	1	1
29	11100	4	0	1	1	0	1	0	1
30	11101	4	0	1	1	0	0	1	1
31	11110	3	0	1	1	0	1	0	0
32	11111	3	0	1	1	0	0	1	0

Search Tree: Hill Climbing



Search Tree: Beam Search (w=2)



Search Tree Nodes

Search tree nodes appear in MoveGen order.

Search tree nodes display up to five pieces of information: node label, **in**spection **o**rder (INO), cost, **op**en **o**rder (OPO) and node status. The first four are printed in the node and the last one (node status) is shown visually by color and line type.

Node label and node status are mandatory information, the remaining are optional.

	CLOSED	OPEN	PRUNED
LABEL;INO COST (OPO)	11111;1 3 (1)	00001 6 (15)	11011 3