Timetabling Problem

	8	9	
sat	(AW, 207, 13A)		
sun			

প্রবলিম্ব number দিয়ে প্রকাশ করব, D0, D1 (not sat, sun)

	P0	P1	
D0	(T1, R1, C1)		
D1	T1, R1, C1 T3, R2, C2		
D2			

Actually T1 store করব না, store করব just 1.

3 conflicts.

① Teacher

② Room

③ class

In (D_1, P_0):

conflict of T_1 :

এ ঘরে কী কোন T_1 এর element আছে, থাকলে কয়টা আছে।

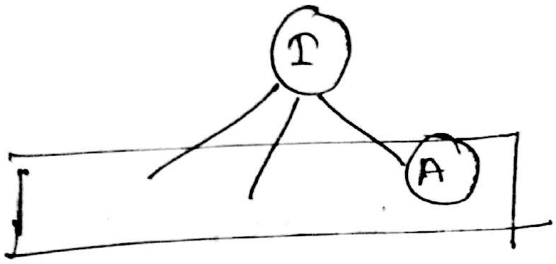
$= 2 - 1 = 1$

conflict of $T_2 = 0$

" " $T_3 = 1 - 1 = 0$

* All possible timetable, each of them is node

একটা Action perform করলে যেটা হবে সেটার neighbor.



(period-1) total successors.

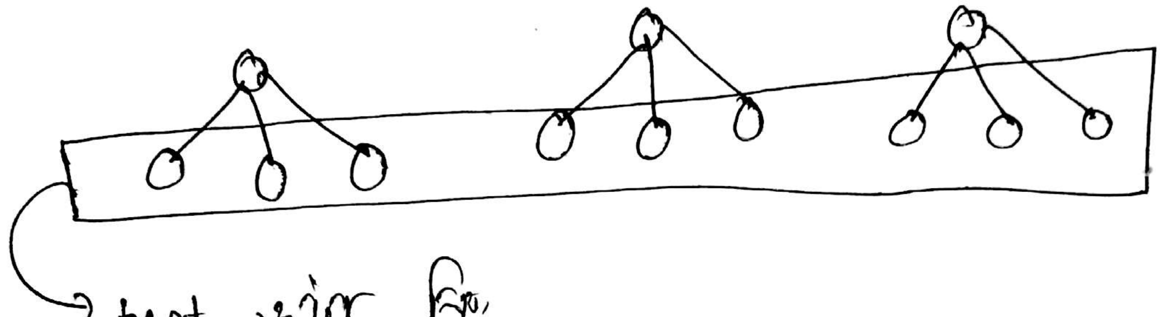
successor এর মধ্যে যদি best টা parent এর চেয়ে best হলে, তাহলেই আবার যাব, নইলে না,

Hill climbing:

```
state s = getInitState();      এটা file read করে একটা state বানাবে.
while true(){
    PriQ = new PriorityQueue()
    PriQ.addAll( s.getSuccessors; )   [linked list এর চেয়ে effecient but priority queue use করব, code কম লিখতে লাগবে]
    best = PriQ.remove()
    if (s.cost ≤ best cost)
        return s
    s = best
}
```

Local beam search:

ক random কয়টি initial state



→ best 3টা নিব,

Local beam search:

```
state [] s = getInitstates()
```

→ sort করে আমরা state array তে রাখতে পারি

```
while true ()
{
    PriQ = new PriorityQ

    for i=0 to (k-1):
        PriQ. addALL (s[i].getSuccessors())

    best = priQ.remove().

    if s[0] cost ≤ best cost
        returns s[0]

    s[0] = best
```

```
for  i=1  to (k-1)
        S[i] = priQ.remove( )
    }
}
```

<u>Stochastic:</u>

just best না নিয়ে, randomly নিই,

~~[illegible]~~

```
state[] S = getInitstate( )
While (true)
{       L = new ArrayList ( )
        for  i = 0  to (k-1) {
L.add ALL = ( S[i] . get successors ( );) }.

max Cost = ........

L' = new ArrayList ( )
```

~~for each~~

for each i=0 to (L.size)

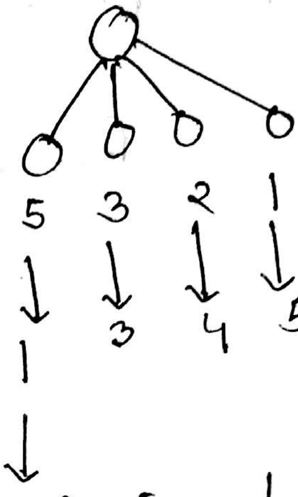
n = (maxCost + 1) - x.cost

for j=0 to n-1

L'.add (i)

. . . .

}



(5+1) - 5 = 1.

(5+1) - 3 = 3.

(5+1) - 2 = 4.

* যদি অনেকক্ষণ চলে, তাহলে return করে দিব অথবা fixed number of iterations

Random restart:

কয়েকবার আবার random restart

getInitState().

// Add 4 note

number of Teachers.

n Classes.

n Room

```
for r = 0 to (nRoom - 1)
    for c = 0 to (nClass - 1)
        for t = 0 to (nTeachers - 1)

            x = sconf
            for (i = 0 to (x - 1))
                Random number ~~of periods~~ from
                    total number of
                    periods.
                then place this element combination
                    in this period.

getSuccessors()
    List =
    for each period P1
        for each element e ∈ P1
            for each period, P2
                if P1 ≠ P2.
                    create a new state that
                    is identical to the state.
```

~~expect~~ such that P_1 doesn't contain P_2 does.

P_1 থেকে remove করে P_2 তে বসাবো.