20CYS304 AI&NN

Unit 1

Problem solving using Hill Climbing Algorithm

Note. Refer class notes "Unit1 _part 3" for Hill climbing algorithm and other examples

Example

In a firewall, multiple rules determine what traffic is allowed or denied. These rules can become inefficient over time, either due to redundancy or suboptimal ordering (as firewalls usually process rules sequentially). Poorly ordered rules increase latency and reduce the effectiveness of the firewall.

Given a list of N firewall rules, find the optimal ordering of these rules to minimize the average packet processing time, assuming each rule has a different match frequency.

Let:

- R = [r1, r2, ..., rn] be the list of firewall rules.
- Each rule ri has a **match frequency** fi (how often it is triggered).
- The **cost** of a rule at position i is: cost(ri) = fi × i
- The **goal** is to reorder the rules such that the **total cost** (sum of all rule costs) is minimized.

Suppose you have four firewall rules,

Rule	Match Frequency (fi)
R1	50
R2	20
R3	10
R4	5

How do you use **hill climbing** to iteratively improve the ordering?

Solution:

Steps:

1) **Start with an initial ordering** (random or frequency-based).
2) **Evaluate total cost**.
3) **Generate neighbors** by swapping two rules.
4) **Move to the neighbor** with the lowest cost.
5) **Repeat** until no better neighbor exists.

Initial Order: [R1, R2, R3, R4]

Initial Cost:

- R1: 50×1 = 50
- R2: 20×2 = 40
- R3: 10×3 = 30
- R4: 5×4 = 20
- **Total = 140**

First Iteration (Swap R1 and R2):

New Order: [R2, R1, R3, R4]

New Cost:

- R2: 20×1 = 20
- R1: 50×2 = 100
- R3: 10×3 = 30
- R4: 5×4 = 20
- **Total = 170** ❌(Worse)

Try [R1, R3, R2, R4] → Cost = 145 ❌

Try [R1, R2, R4, R3] → Cost = 135 ☑less than the initial order cost.

Move to [R1, R2, R4, R3]

Repeat..

Eventually, hill climbing leads to:

Optimal Order: [R4, R3, R2, R1]
Total Cost:

- 5×1 + 10×2 + 20×3 + 50×4 = 5 + 20 + 60 + 200 = 285 ❌Too high
  Better is: [R1, R2, R3, R4] = 140 ☑

Even better: [R1, R3, R2, R4] = 135 ☑☑

Final result might be [R1, R3, R2, R4]

The hill climbing algorithm gives a locally optimal rule order that minimizes processing cost, improving packet filtering efficiency without redesigning the firewall.

This method can be applied to:

- Intrusion detection rule prioritization (e.g., Snort signatures)
- Signature order in antivirus scanning
- Log analysis filter sequence optimization