## Step 1 Create pattern table

a) Containing patterns of non-critical code

Note: We only consider “related” codes to form patterns; Related is defined as codes occurred < 1 week before. (***one week window time pattern***)

b) Mark it whether or not a critical code followed within 1 week

Critical A B Critical B A C C Critical

For Veh\_i

1. Find non-critical codes, denote the set as {F\_J}
2. Of {F\_J}, find codes that has next critical code coming not less than 60 mins , denotes the set as {F\_K}
3. For each f\_k in {F\_K}
   1. Check whether there is a critical code coming in 1 week
   2. Find “related” codes set {f\_k}
      1. They must occurred <1 week before f\_k
      2. They must after proceeding critical code
4. Build Table T1 and T2 for each f\_k

**T1: Related codes to form patterns:** for non-critical code without critical code coming in next 60 mins, list all non-critical codes in proceeding 1-week window (truncated at proceeding critical code)

|  |  |  |  |
| --- | --- | --- | --- |
| End Code ID (f\_k) | Preceding Code ID | Code | TimeStamp |
| 1 | 1 | A |  |
| 2 | 1 | A |  |
| 2 | 2 | B |  |
| 3 | 3 | B |  |
| 4 | 3 | B |  |
| 4 | 4 | A |  |
| 5 | 3 | B |  |
| 5 | 4 | A |  |
| 5 | 5 | C |  |
| 6 | 3 | B |  |
| 6 | 4 | A |  |
| 6 | 5 | C |  |
| 6 | 6 | C |  |

**T2: Pattern(1-week window) table index by end code**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| End Code ID | Critical Code in 1 week | # fault in pattern | Pattern | Veh\_ID | Start Time | End Time |
| 1 |  |  | A |  |  | 10 |
| 2 |  |  | AB |  |  | 12 |
| 3 |  |  | B |  |  | 14 |
| 4 |  |  | BA |  |  | 16 |
| 5 |  |  | BAC |  |  | 18 |
| 6 |  |  | BACC |  |  | 20 |

## Step 2 Create pattern and sub pattern table

1. Identify distinct patterns in T2(Across all vehs), denote as set {P\_I}
2. For each pattern in T2, identify its sub patterns in {P\_I} and create T3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| End Code ID | Critical Code in 1 week | # fault in pattern | Pattern | Veh\_ID | Start Time | End Time |
| 1 |  | 1 | A |  | 10 | 10 |
| 2 |  | 2 | A |  | 10 | 12 |
| 2 |  |  | B |  |  |  |
| 2 |  |  | AB |  |  |  |
| 3 |  |  | B |  |  |  |
| 4 |  | 2 | A |  | 14 | 16 |
| 4 |  |  | B |  |  |  |
| 4 |  |  | BA |  |  |  |
| 5 |  | 3 | A |  | 14 | 1 |
| 5 |  |  | B |  |  |  |
| 5 |  |  | BA |  |  |  |
| 5 |  |  | BAC |  |  |  |
| 6 |  |  | A |  |  |  |
| 6 |  |  | B |  |  |  |
| 6 |  |  | BA |  |  |  |
| 6 |  |  | BAC |  |  |  |
| 6 |  |  | BACC |  |  |  |

patterns.csv 🡺 pat\_and\_sub.csv

1. Find distinct patterns from patterns.csv and denote as **uniq\_pat=**{Pattern1, Pattern 2….}
2. Find unique component (building block) from **uniq\_pat** anddenote as **pat\_comp**={comp1,…,compK}
3. Build component count table for each pattern from **uniq\_pat** and denote the set as **uniq\_pat\_comp\_n**

Examples

Pattern 1: count\_T1

|  |  |
| --- | --- |
| Component | Count |
| Comp1 | 0 |
| Comp2 | 2 |
| Comp3 | 3 |
| … |  |
| CompK | 1 |

Pattern 2: count\_T2

|  |  |
| --- | --- |
| Component | Count |
| Comp1 | 4 |
| Comp2 | 1 |
| Comp3 | 0 |
| … |  |
| CompK | 0 |

1. Find the sub-pattern of each pattern from patterns.csv
   1. Given a pattern from patterns.csv, build its component count table

pattern\_count\_T

|  |  |
| --- | --- |
| Component | Count |
| Comp1 | 2 |
| Comp2 | 0 |
| Comp3 | 0 |
| … |  |
| CompK | 0 |

* 1. Find its sub-patterns by comparing pattern\_count\_T with count\_T1, count\_T2,….

If the count in pattern\_count\_T >= count in count\_Ti for every components, then pattern i from **uniq\_pat** will be regarded as one of the sub-patterns.

1. Remove any duplicates from T3.
   1. Keys: veh\_id, pattern, overlapping timestamp
   2. With the same keys in a), remove records with fewer faults

## Step 3 Analysis T3