حوری دهش 9821413

با مثال نشان دهید چرا در تئوری شرط infeasible نبودن TRها، برای رابطه C1 subsumbes C2 الزامی است؟

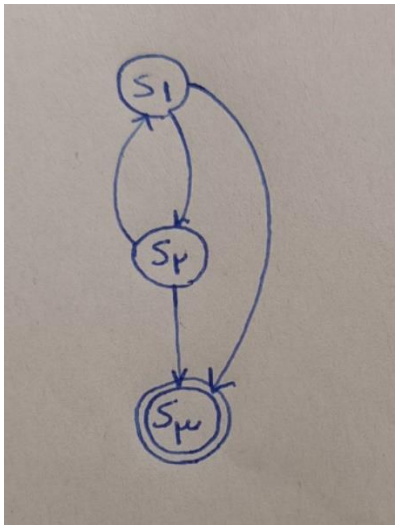
توضیح عملی:

در حالت کلی اگر:

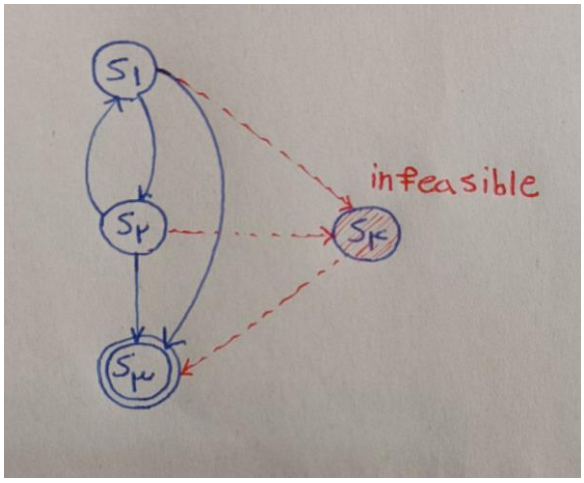
معیار پوشش C1 : هر یال باشد

معیار پوشش C2 : هر نود یا گره باشد

در حالت کلی اگر درون TR ها infeasible وجود نداشته باشد معیار پوشش C1 معیار پوشش C2 را subsumes میکند. شکل زیر برای این توضیح در نظر گرفته شده است:




اگر درون TR ها infeasible وجود داشته باشد --> روی شکل زیر هم کد و هم توضیح را انجام داده ام:



معیار پوشش C1 : هر یال باشد

معیار پوشش C2 : هر نود یا گره باشد

در حالت کلی مسیرهایی که در گراف بالا داریم به صورت زیر هستند:

{S1S2 , S2S1 , S1S3 , S2S3}

تست requirement برای C1 :

TR1: {S1S2 , S2S1 , S1S3 , S2S3 , S1S4 , S2S4 , S4S3}

تست requirement برای C2 :

TR2: {S1 , S2 , S3 , S4}

درون TR1 چندتا infeasible وجود دارد که با قرمز نشان داده شده است و درون TR2 هم یک infeasible وجود دارد پس در نتیجه نمی توان درون تست ست، تست کیسی یا تست کیس هایی طراحی کرد که کل TR1 را satisfy بکند. در این حالت معیار پوشش C1 معیار پوشش C2 را نمی تواند subsumes کند پس برای همین است که گفته میشود درون TRها نباید infeasible وجود داشته باشد.

کد جاوا برای این مثال:

```
import java.util.*;

class Node {
    String name;
    List<Node> neighbors;

    Node(String name) {
        this.name = name;
        this.neighbors = new ArrayList<>();
    }

    void addNeighbor(Node neighbor) {
        neighbors.add(neighbor);
    }
}
```

```
public class GraphExample {

    static boolean isFeasible(Node node) {
        return !node.name.equals("S4");
    }

    public static void main(String[] args) {
        Node S1 = new Node("S1");
        Node S2 = new Node("S2");
        Node S3 = new Node("S3");
        Node S4 = new Node("S4");    //infeasible

        S1.addNeighbor(S2);
        S1.addNeighbor(S3);
        S2.addNeighbor(S1);
        S2.addNeighbor(S3);

        List<Node> nodes = Arrays.asList(S1, S2, S3, S4);

        for (Node node : nodes) {
            if (isFeasible(node)) {
                System.out.println("Node " + node.name + " is feasible.");
                for (Node neighbor : node.neighbors) {
                    if (isFeasible(neighbor)) {
                        System.out.println("  -> Can transition to " + neighbor.name);
                    } else {
                        System.out.println("  -> Transition to " + neighbor.name + " is infeasible.");
                    }
                }
            } else {
                System.out.println("Node " + node.name + " is infeasible.");
            }
        }
    }
}
```

توضیح تئوری:

اگر درون TRهای معیار پوشش C1، infeasible وجود داشته باشد این به این معناست که این تست ها قابلیت اجرا یا اجرای موثری ندارند پس نمی توانند حتی پوشش C1 را فراهم کنند چه برسد به پوشش C2

مطابق با برهان خلف می توانیم این موضوع را اثبات کنیم:

فرض کنید که معیار پوشش C1 معیار پوشش C2 را subsumes می کند و در عین حال تست هایی برای C1 وجود دارد که infeasible هستند.

اگر معیار پوشش C1 معیار پوشش C2 را subsumes کند هر تستی که برای C1 طراحی شده است پوشش C2 را نیز فراهم می کند.

اما فرض کردیم که درون TR1 یکسری infeasible برای C1 وجود دارد بنابراین نمیتوان درون تست ست، تست کیس یا تست کیس هایی طراحی کرد که کل TR1 را satisfy کند پس این تست ها حتی نمی توانند به درستی پوشش C1 را فراهم کنند بنابراین نمی‌توان انتظار داشت که پوشش C2 به درستی فراهم شود.

این با فرض ما که معیار پوشش C1 معیار پوشش C2 را subsumes می‌کند در تناقض است.

در نتیجه اگر درون TR1 برای معیار پوشش C1 یکسری infeasible وجود داشته باشد، معیار پوشش C1 نمی تواند معیار پوشش C2 را subsumes کند پس تئوری شرط infeasible نبودن TRها، برای رابطه C1 subsumbes C2 الزامی است.

حالا اگر درون TR1 یکسری infeasible برای معیار پوشش C1 وجود داشته باشد و معیار پوشش C1 معیار پوشش C2 را subsumes بکند در این حالت یکسری مشکلاتی ممکن است به وجود آید از قبیل:

- نقص در پوشش تست ها: اگر تست های موجود برای رسیدن به پوشش C2 طراحی نشده باشند، این به معنای این است که بخشی از کد سیستم تست نشده و آسیب پذیر می ماند.
- کاهش اعتماد به نرم‌افزار: وجود خطاهایی که توسط تست های موجود شناسایی نشده‌اند، میتواند باعث کاهش اعتماد به سیستم شود. این میتواند به مشکلات امنیتی، عملکردی یا قابلیت اطمینان مربوط شود.
- افزایش ریسک: عدم تست و پوشش کامل سیستم باعث افزایش ریسک است. اگر اشکالاتی وجود داشته باشد که توسط تست ها شناسایی نشده‌اند، ممکن است در محیط عملیاتی سیستم منجر به مشکلات جدی شود.
- مشکلات قابل شناسایی نشدن: بدون تست های مناسب، مشکلات در سیستم ممکن است تا زمانی که در محیط عملیاتی پدیدار شوند شناسایی نشوند. این میتواند باعث افزایش هزینه ها و زمان های تعمیر و نگهداری شود.