Chương 4

Kiểm thử hộp đen

(Buổi 7, 8, 9)

Nội dung

- Kiểm thử hộp đen là gì?
- Kỹ thuật phân chia lớp tương đương (*equivalence class partitioning*)
- Kỹ thuật phân tích giá trị biên (*boundary value analysis*)
- Kỹ thuật phân tích miền (*domain analysis*)
- Kỹ thuật bảng quyết định (*decision table*)
- Kỹ thuật cặp đôi (*pairwaise*)
- Kỹ thuật chuyển trạng thái (*state transition*)
- Kỹ thuật đồ thị nhân quả (*cause-effect graph*)
- Kỹ thuật trường hợp sử dụng (*use-case*)

Kiểm thử hộp đen

- Trong **kiểm thử hộp đen** (***black-box tetsting***), phần mềm được xem là một hộp đen và các *test-case* được xác định từ các đặc tả yêu cầu chức năng của thành phần phần mềm, không dựa vào mã nguồn của thành phần phần mềm.

Kiểm thử hộp đen

❖ Các kỹ thuật kiểm thử hộp đen

- Kỹ thuật phân chia lớp tương đương (*equivalence class partitioning*)
- Kỹ thuật phân tích giá trị biên (*boundary value analysis*)
- Kỹ thuật phân tích miền (*domain analysis*)
- Kỹ thuật bảng quyết định (*decision table*)
- Kỹ thuật cặp đôi (*pairwaise*)
- Kỹ thuật chuyển trạng thái (*state transiton*)
- Kỹ thuật nhân quả (*cause-effect*)
- Kỹ thuật trường hợp sử dụng (*use-case*)

Qui trình kiểm thử hộp đen

- **Bước 1:** Phân tích các đặc tả yêu cầu chức năng của thành phần phần mềm.
- **Bước 2:** Xác định các *test-case*. Thông tin của mỗi *test-case* gồm:
  - Dữ liệu nhập (giá trị hợp lệ, giá trị không hợp lệ).
  - Trạng thái hiện tại của thành phần phần mềm.
  - Kết quả mong muốn khi chạy thành phần phần mềm.
- **Bước 3:** Chạy các *test-case* và kiểm tra kết quả so với kết quả mong muốn.
- **Bước 4:** Lập báo cáo kết quả kiểm thử để phản hồi cho những người liên quan.

Kỹ thuật phân chia lớp tương đương

- **Kỹ thuật phân chia lớp tương đương (*equivalence class partitioning*) phân chia các giá trị của dữ liệu nhập thành các nhóm dữ liệu, mỗi nhóm dữ liệu là một lớp tương đương.**
- **Xác định một *test-case* cho một lớp tương đương.**
- **Số lượng *test-case* của thành phần phần mềm được giảm đáng kể, chất lượng kiểm thử vẫn được chấp nhận.**

Kỹ thuật phân chia lớp tương đương

- **Lớp tương đương** (*equivalence class*) bao gồm các giá trị mà khi chạy một thành phần phần mềm với các giá trị này thì nó thực hiện cùng một hành vi.
  - Khi chạy một thành phần phần mềm với một giá trị nào đó thuộc một lớp tương đương và bị lỗi sai, thì vẫn bị lỗi sai này với các giá trị khác thuộc cùng lớp tương đương này.
  - Khi chạy một thành phần phần mềm với một giá trị nào đó thuộc một lớp tương đương và không bị lỗi sai, thì vẫn không bị lỗi sai này với các giá trị khác thuộc cùng lớp tương đương này.

Kỹ thuật phân chia lớp tương đương

- **Các cách kiểm thử:**
  - **Cách 1: Kiểm thử theo hợp đồng** (TbC – *testing by contract*)
    - **Thiết kế theo hợp đồng** (*Design by contract* – DbC) là cách tiếp cận để thiết kế phần mềm, người thiết kế phần mềm phải xác định các đặc tả giao tiếp chính thức, chính xác và có thể kiểm chứng cho các thành phần phần mềm.
    - **Hợp đồng** (*contract*) phát biểu các điều mà cả hai bên phải làm, không phụ thuộc vào cách thức thực hiện. Hợp đồng thường được xác định bởi các khẳng định và các khái niệm liên quan.

Kỹ thuật phân chia lớp tương đương

- **Khẳng định** (***assertion***) là một biểu thức luận lý liên quan đến một số thực thể của phần mềm và nêu ra đặc tính mà các thực thể này có thể đáp ứng ở những giai đoạn nào đó khi thực hiện phần mềm.
- Có ba loại khẳng định:

Điều kiện trước (***pre-condition***): điều kiện phải được thỏa mãn trước khi thực hiện phương thức, nó liên quan đến trạng thái của hệ thống và các đối số được truyền cho phương thức.

Điều kiện sau (***post-condition***): điều kiện phải được thỏa mãn sau khi thực hiện phương thức.

Bất biến (***invariant***): điều kiện phải được thỏa mãn ở mọi thời điểm gọi phương thức, được kiểm tra trước và sau khi thực hiện phương thức. Vi phạm một khẳng định có thể cho thấy một lỗi sai của thành phần phần mềm.

Kỹ thuật phân chia lớp tương đương

- **Cách 2: Kiểm thử phòng vệ (*defensive testing*)**
  - **Lập trình phòng *vệ* (*Defensive programming*) là dạng thiết kế phòng vệ nhằm bảo đảm chức năng của thành phần phần mềm trong các trường hợp chưa biết trước, cách thiết kế này được sử dụng khi thành phần phần mềm có thể được sử dụng sai.**
  - **Lập trình phòng vệ là cách tiếp cận để cải tiến phần mềm và mã nguồn về phương diện:**

Giảm số lượng các lỗi sai của phần mềm.

Làm cho mã nguồn dễ đọc, dễ hiểu và giúp ích cho kiểm thử hộp trắng.

Làm cho phần mềm chạy theo một cách thức biết trước với dữ liệu nhập bất kỳ (hợp lệ và không hợp lệ).

Kỹ thuật phân chia lớp tương đương

- Tuy nhiên, lập trình phòng vệ có thể dẫn đến việc tồn tại mã nguồn xử lý các lỗi sai không thể xảy ra nhưng vẫn được thực hiện trong thời gian chạy, kích khởi rất nhiều trường hợp ngoại lệ (*expception*). Điều này làm tăng thời gian chạy và chi phí bảo trì phần mềm.
- Xác định các lớp tương đương cho các giá trị không hợp lệ.

Kỹ thuật phân chia lớp tương đương

- Ví dụ: Đặc tả yêu cầu chức năng tuyển dụng nhân viên như sau phụ thuộc vào tuổi của ứng viên xin việc (dữ liệu nhập).

Tuổi	Hành vi
Từ 0 đến dưới 16	Không tuyển
Từ 16 đến dưới 18	Tuyển bán thời gian
Từ 18 đến dưới 55	Tuyển toàn thời gian
Từ 55 đến 99	Không tuyển

Kỹ thuật phân chia lớp tương đương

- **Cách 1:** kiểm thử **tất cả các giá trị hợp lệ**, có 100 *test-case* (tuổi: 0, 1, 2, …, 99).
- **Cách 2:** kiểm thử **tất cả các lớp tương đương**, có 4 lớp tương, mỗi lớp tương đương có một *test-case*:
  - **Lớp 1: tuổi từ 0 đến 16**
    - *Test-case* T1: nhập: 12 xuất: không tuyển
  - **Lớp 2: tuổi từ 16 đến 18**
    - *Test-case* T2: nhập: 17 xuất: tuyển bán thời gian
  - **Lớp 3: tuổi từ 18 đến 55**
    - *Test-case* T3: nhập: 40 xuất: tuyển toàn thời gian
  - **Lớp 4: tuổi từ 55 đến 99**
    - *Test-case* T4: nhập: 90 xuất: không tuyển

Kỹ thuật phân chia lớp tương đương

- Xét đoạn mã lệnh với dữ liệu nhập của tuổi từ 0 đến 99:

```
if (tuoi >= 0 && tuoi < 16)     kq = "Khong tuyen";
if (tuoi >= 16 && tuoi < 18)    kq = "Tuyen ban thoi gian";
if (tuoi >= 18 && tuoi < 55)    kq = "Tuyen toan thoi gian";
if (tuoi >= 55 && tuoi <= 99)   kq = "Khong tuyen";
```

- Kiểm thử hộp trắng với các *test-case* T1, T2, T3, T4:
  - Tất cả các phát biểu đã được kiểm thử (100%).
  - Tất cả các đường độc lập tuyến tính cơ bản đã được kiểm thử (100%).

Kỹ thuật phân chia lớp tương đương

- Xét đoạn mã lệnh với dữ liệu nhập của tuổi từ 0 đến 99 (100 phát biểu dựa vào các giá trị của dữ liệu nhập, không dựa vào đặc tả yêu cầu chức năng):

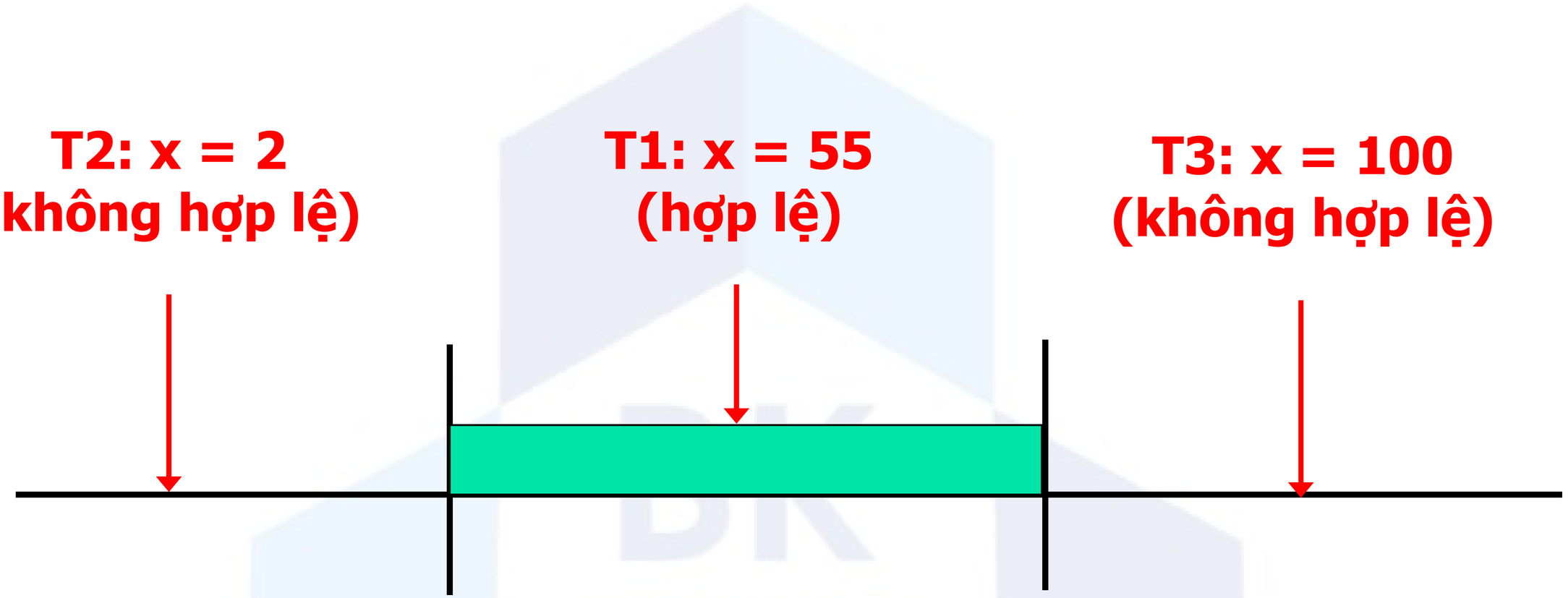
```
if (tuoi == 0)   kq = "Khong tuyen"; …
if (tuoi == 12) kq = "Khong tuyen"; …
if (tuoi == 17) kq = "Tuyen ban thoi gian"; …
if (tuoi == 40) kq = "Tuyen toan thoi gian"; …
if (tuoi == 90) kq = "Khong tuyen";
if (tuoi == 99) kq = "Khong tuyen";
```

- Kiểm thử hộp trắng với các *test-case* T1, T2, T3, T4:
  - 4 / 100 phát biểu đã được kiểm thử (4%).
  - 4 đường độc lập tuyến tính cơ bản đã được kiểm thử (4%).

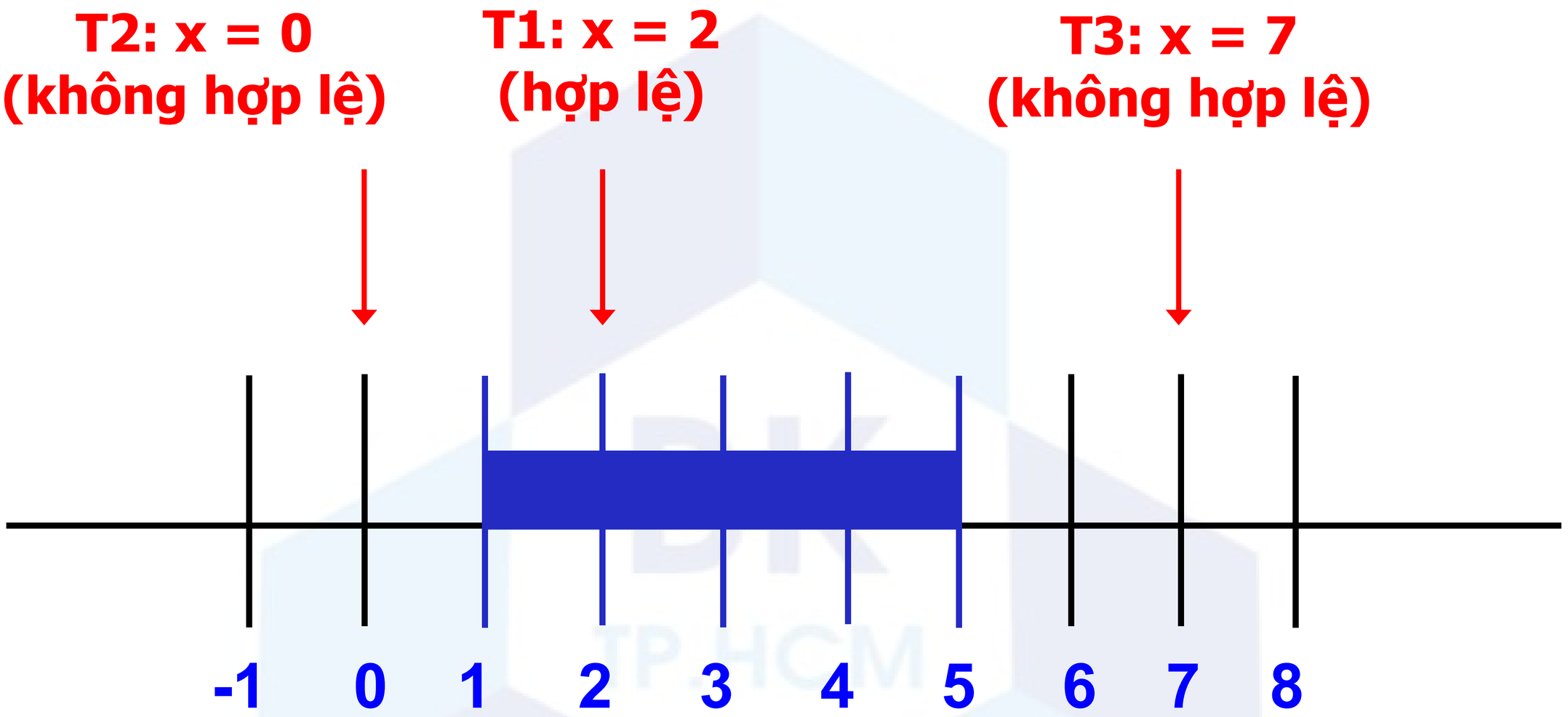
Kỹ thuật phân chia lớp tương đương

- **Dữ liệu nhập thuộc một miền trị liên tục hoặc thuộc một miền trị rời rạc có thứ tự.**
  - **Một lớp tương đương bao gồm các giá trị hợp lệ, ví dụ đoạn [*m*, *n*] với *m* ≤ *n*.**
    - **Một *test-case* T1: *m* < *x* < *n*.**
  - **Hai lớp tương đương bao gồm các giá trị không hợp lệ.**
    - **Một lớp tương đương bao gồm các giá trị nhỏ hơn *m*. Một *test-case* T2: *x* < *m*.**
    - **Một lớp tương đương bao gồm các giá trị lớn hơn *n*. Một *test-case* T3: *x* > *n*.**

Kỹ thuật phân chia lớp tương đương



Kỹ thuật phân chia lớp tương đương



Kỹ thuật phân chia lớp tương đương

- Dữ liệu nhập thuộc một miền trị rời rạc độc lập nhau $D_1 = \{v_1, v_2, \ldots, v_m\}$.
  - Một lớp tương đương bao gồm các giá trị hợp lệ thuộc D_1.
    - Một *test-case* T1: $x = v_k$ với $v_k \in D_1$.
  - Một lớp tương đương bao gồm các giá trị không hợp lệ $D_2 = \{i_1, i_2, \ldots, i_n\}$.
    - Hai *test-case* T2: $x = i_{k1}$ T3: $x = i_{k2}$ với $i_{k1}, i_{k2} \in D_2$.
    - Nên có thêm các *test-case* tương ứng với các giá trị không hợ lệ còn lại.

Kỹ thuật phân chia lớp tương đương

T1: loaihb = Xuất sắc

T2: loaihb = Trung bình

T3: loaihb = Kém

Loại học bổng

Kỹ thuật phân chia lớp tương đương

- **Kiểm thử nhiều loại dữ liệu nhập.**
  - **Có *n* loại dữ liệu nhập, mỗi loại dữ liệu nhập có các giá trị hợp lệ và các giá trị không hợp lệ.**
    - **Diện chính sách:**

D_{1HL} = {Nghèo, Con liệt sĩ}

D_{1KHL} = {Khó khăn, Vùng xa, Dân tộc}

V_{1HL} = {Nghèo}

V_{1KHL} = {Khó khăn, Vùng xa}

- **Học lực:**

D_{2HL} = {Xuất sắc, Giỏi, Khá}

D_{2KHL} = {Trung bình, Yếu, Kém}

V_{2HL} = {Giỏi}

V_{2KHL} = {Trung bình, Kém}

Kỹ thuật phân chia lớp tương đương

- **Cách 1: Một *test-case* cho mỗi tổ hợp *n* loại dữ liệu nhập.**
  - **Mỗi loại dữ liệu nhập được lấy ra có 3 giá trị (1 giá trị hợp lệ và 2 giá trị không hợp lệ).**
  - **Số lượng *test-case* là 3^n (có thể rất lớn).**

Kỹ thuật phân chia lớp tương đương

Test-case	Diện chính sách	Học lực	Kết quả
T1	Nghèo	Giỏi	Hợp lệ
T2	Nghèo	Trung bình	Không hợp lệ
T3	Nghèo	Yếu	Không hợp lệ
T4	Khó khăn	Giỏi	Không hợp lệ
T5	Khó khăn	Trung bình	Không hợp lệ
T6	Khó khăn	Yếu	Không hợp lệ
T7	Vùng xa	Giỏi	Không hợp lệ
T8	Vùng xa	Trung bình	Không hợp lệ
T9	Vùng xa	Yếu	Không hợp lệ

Kỹ thuật phân chia lớp tương đương

- **Kiểm thử nhiều loại dữ liệu nhập.**
  - **Cách 2: *n* + 2 *test-case***
    - **Một *test-case* cho các giá trị của *n* loại dữ liệu nhập hợp lệ.**
    - **Một *test-case* cho các giá trị của *n* loại dữ liệu nhập không hợp lệ.**
    - ***n test-case* cho các tổ hợp của *n* loại dữ liệu nhập, trong đó có một giá trị hợp lệ (thay đổi cho mỗi loại dữ liệu nhập) và các giá trị còn lại là không hợp lệ.**

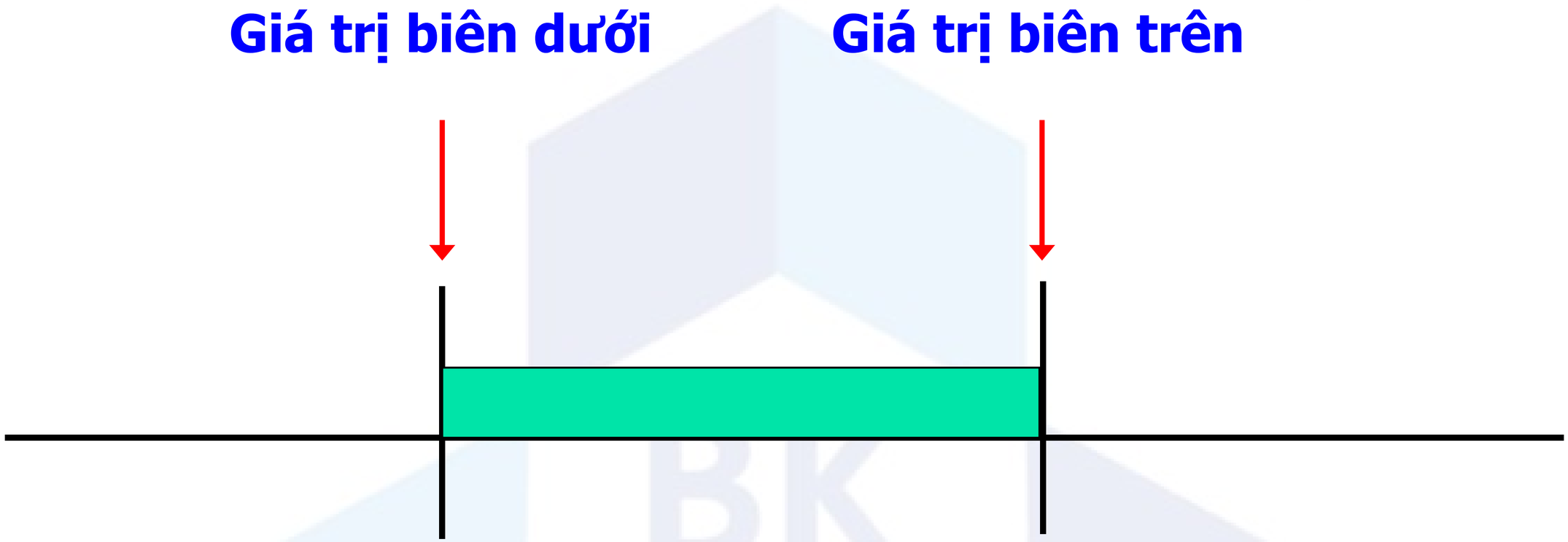
Kỹ thuật phân chia lớp tương đương

Test-case	Diện chính sách	Học lực	Kết quả
T1	Nghèo	Giỏi	Hợp lệ
T2	Vùng xa	Trung bình	Không hợp lệ
T3	Vùng xa	Giỏi	Không hợp lệ
T4	Nghèo	Trung bình	Không hợp lệ

Kỹ thuật phân tích giá trị biên

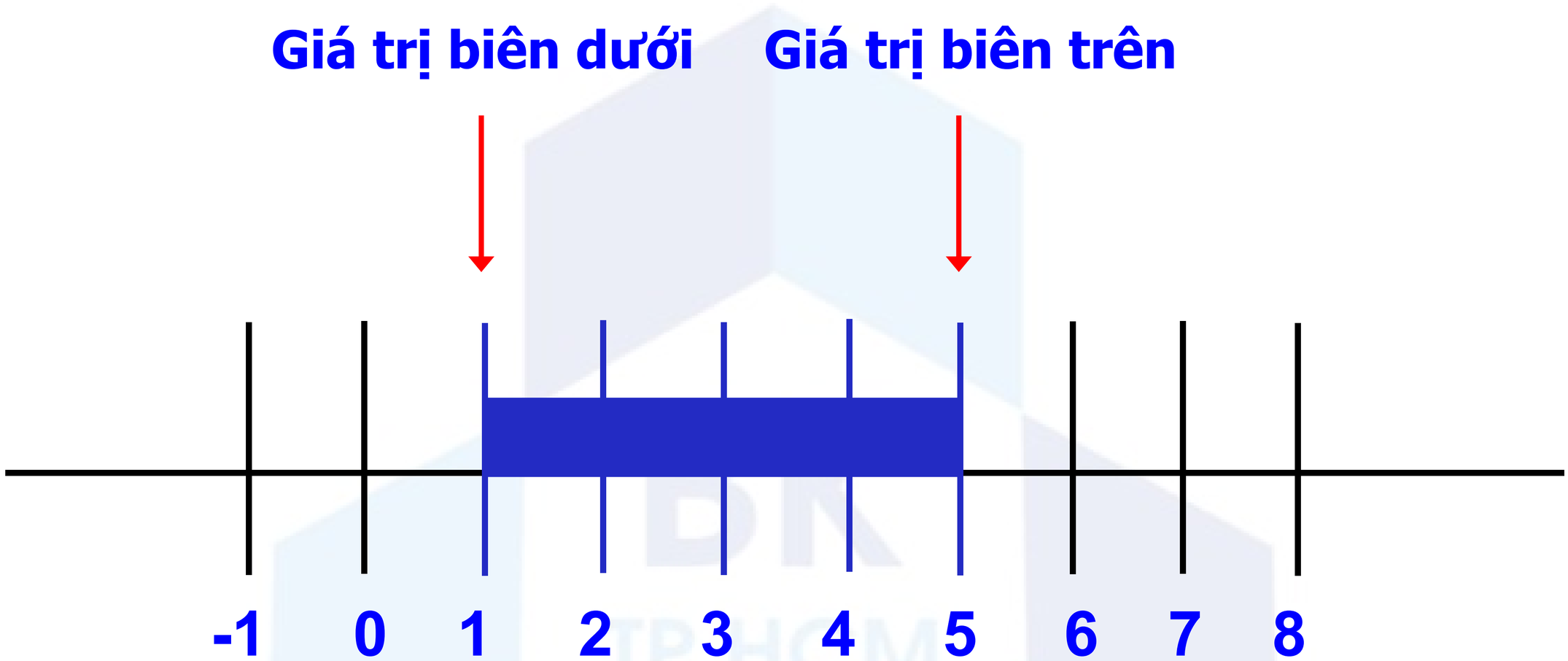
- Lỗi sai thường xảy ra tại các giá trị biên của miền trị.
- Kỹ thuật phân tích giá trị biên (*boundary value*) tập trung thiết kế các *test-case* cho các giá trị biên.
- Để tránh các lỗi sai xảy ra tại các giá trị biên, cần phải kiểm duyệt mã nguồn (*code inspection*).

Kỹ thuật phân tích giá trị biên



Miền trị liên tục

Kỹ thuật phân tích giá trị biên



Miền trị rời rạc có thứ tự

Kỹ thuật phân tích giá trị biên

- **Ví dụ: Đặc tả yêu cầu chức năng tuyển dụng nhân viên như sau phụ thuộc vào tuổi của ứng viên xin việc (dữ liệu nhập).**

Tuổi	Hành vi
Từ 0 đến dưới 16	Không tuyển
Từ 16 đến dưới 18	Tuyển bán thời gian
Từ 18 đến dưới 55	Tuyển toàn thời gian
Từ 55 đến 99	Không tuyển

Kỹ thuật phân tích giá trị biên

- Xét đoạn mã lệnh sau:

```
if (tuoi >= 0 && tuoi <= 16)   { … };
if (tuoi >= 16 && tuoi <= 18) { … };
if (tuoi >= 18 && tuoi <= 55) { … };
if (tuoi >= 55 && tuoi <= 99) { … };
```

- Lỗi sai có thể xảy ra với các giá trị biên 16, 18 và 55. Ta kiểm duyệt mã nguồn sẽ phát hiện một số lệnh bị sai ở toán tử so sánh và viết lại mã nguồn như sau:

```
if (tuoi >= 0 && tuoi < 16)    { … };
if (tuoi >= 16 && tuoi < 18)   { … };
if (tuoi >= 18 && tuoi < 55)   { … };
if (tuoi >= 55 && tuoi <= 99) { … };
```

Kỹ thuật phân tích giá trị biên

- Thiết kế các *test-case* cho các giá trị tại biên và lân cận của biên của các lớp tương đương.
  - **Điều kiện:** lớp tương đương bao gồm các giá trị thuộc một miền trị liên tục hoặc thuộc một miền trị rời rạc có thứ tự.

Kỹ thuật phân tích giá trị biên

- **Bước 1:** Phân tích đặc tả yêu cầu chức năng và xác định các lớp tương đương.
- **Bước 2:** Xác định 2 biên của mỗi lớp tương đương.
- **Bước 3:** Mỗi biên *m* của mỗi lớp tương đương có 3 *test-case*:
  - T1: $x = m$
  - T2: $x < m$
  - T3: $x > m$

Kỹ thuật phân tích giá trị biên

Tuổi	Hành vi
Từ 0 đến 15	Không tuyển
Từ 16 đến 17	Tuyển bán thời gian
Từ 18 đến 54	Tuyển toàn thời gian
Từ 55 đến 99	Không tuyển

- Các *test-case* tương ứng với các giá trị biên: {-1, 0, 1}, {14, 15, 16}, {15, 16, 17}, {16, 17, 18}, {17, 18, 19}, {53, 54, 55}, {54, 55, 56}, {98, 99, 100}.
- Kiểm tra các *test-case* với các tuổi: -1, 0, 1, 14, 15, 16, 17, 18, 19, 53, 54, 55, 56, 98, 99, 100.

Kỹ thuật phân tích giá trị biên

- Kiểm thử *n* lớp tương đương của dữ liệu nhập.
  - Cách 1: Một *test-case* cho mỗi tổ hợp *n* lớp tương đương.
    - Mỗi lớp tương đương có 2 giá trị biên, mỗi giá trị biên có 3 *test-case*.
    - Số lượng *test-case* là 6^n (có thể rất lớn).
  - Cách 2: Một *test-case* cho mỗi tổ hợp *n* loại dữ liệu nhập.
    - Một số *test-case* cho các tổ hợp các giá trị biên của *n* lớp tương đương.
    - Một số *test-case* cho các tổ hợp các giá trị nhỏ hơn hoặc lớn hơn giá trị biên của *n* lớp tương đương.

Kỹ thuật phân tích miền

- Kỹ thuật phân chia lớp tương đương và kỹ thuật phân tích giá trị biên không quan tâm đến mối quan hệ giữa các dữ liệu nhập.

Kỹ thuật phân tích miền

- Thông thường, các dữ liệu nhập có liên quan với nhau, được thể hiện qua các biến dữ liệu có mối quan hệ với nhau.
  - Giá trị của biến này ràng buộc một số giá trị của biến kia.
  - Số lượng các *test-case* là các tổ hợp của các biến này độc lập với nhau sẽ rất lớn.
    - Không thể kiểm thử tất cả các *test-case*.
    - Một số *test-case* không cần thiết (không thể xảy ra).
    - Kiểm thử một số *test-case* nào đó có thể không phát hiện một số lỗi sai có liên quan đến các ràng buộc giữa các biến.

Kỹ thuật phân tích miền

- **Kỹ thuật phân tích miền** (***domain analysis***) xác định các *test-case* khi các biến dữ liệu nhập có liên quan với nhau.
- Trong trường hợp hai biến dữ liệu *x* và *y* có liên quan với nhau, có thể có các lỗi sai:
  - Dịch chuyển biên (*shifted boundary*) theo chiều ngang hoặc dọc
  - Quay nghiêng biên (*tilted boundary*)
  - Thiếu biên (*missing boundary*)
  - Thừa biên (*extra boundary*)

Kỹ thuật phân tích miền

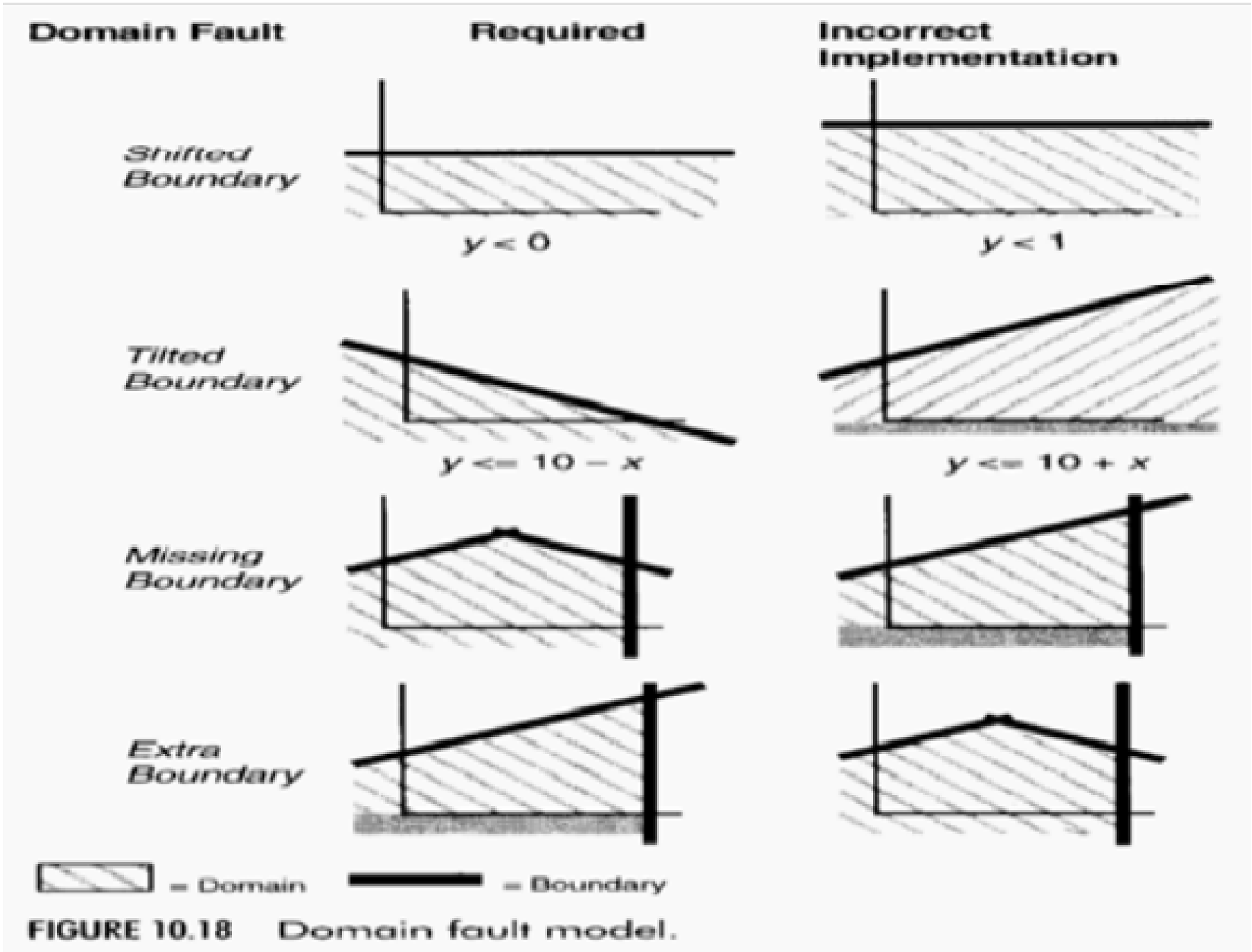


FIGURE 10.18 Domain fault model.

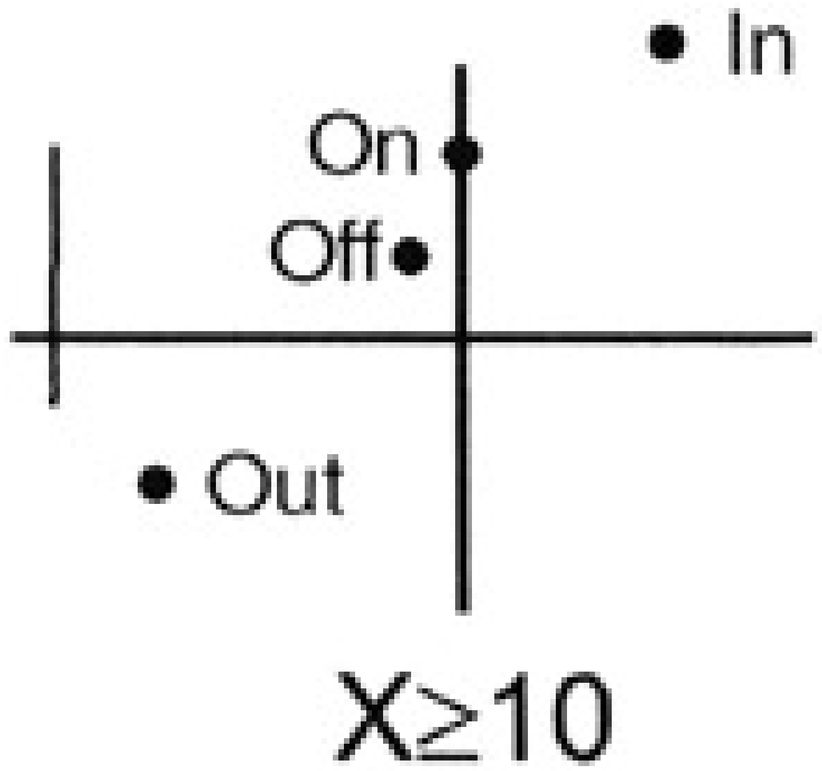
Kỹ thuật phân tích miền

- Các loại điểm
  - Điểm **On** là giá trị thuộc biên.
  - Điểm **Off** là giá trị không thuộc một biên.
  - Điểm **In** là giá trị thỏa mãn tất cả các điều kiện biên nhưng không thuộc một biên.
  - Điểm **Out** là giá trị không thỏa mãn bất kỳ điều kiên biên và không thuộc một biên.

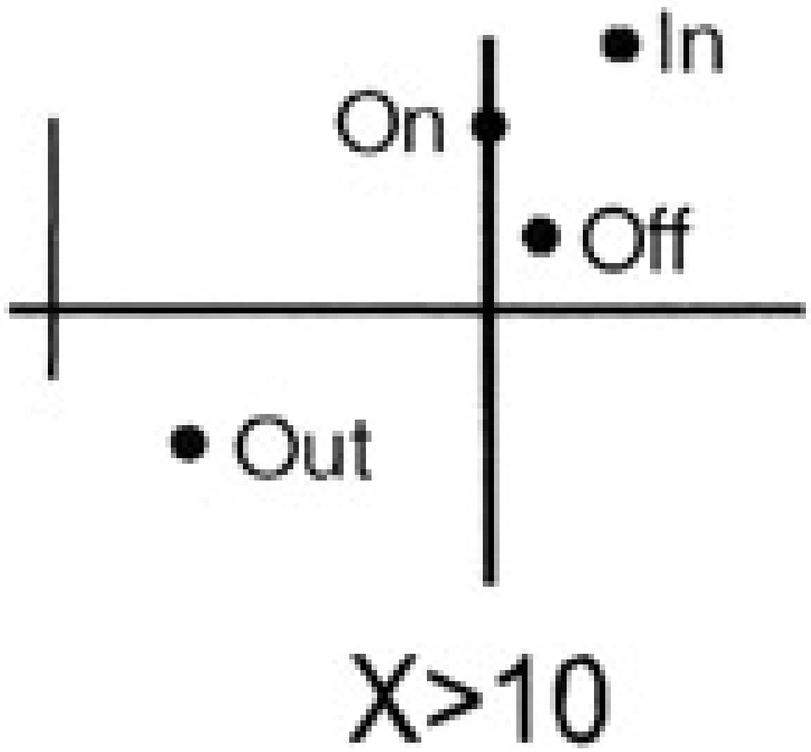
Kỹ thuật phân tích miền

- Chọn các điểm On và Off
  - Đối với biên đóng (*closed boundary*)
    - Được xác định bởi các toán tử =, ≤ hoặc ≥, ví dụ x ≥ 10
    - Các điểm thuộc một biên thì thuộc miền.
    - Điểm On là điểm thuộc biên và thuộc miền, ví dụ x = 10.
    - Điểm Off nằm ngoài miền, ví dụ x = 9.
  - Đối với biên mở (*open boundary*)
    - Được xác định bởi các toán tử < hoặc >, ví dụ x > 10.
    - Điểm On là điểm thuộc biên nhưng không thuộc miền, ví dụ x = 10.
    - Điểm Off là điểm phải thuộc miền, ví dụ x = 11.

Kỹ thuật phân tích miền



X≥10



X>10

Kỹ thuật phân tích miền

- **Tiêu chí chọn *one-by-one***
  - **Đối với điều kiện <, ≤, >, ≥, chọn:**
    - **Một điểm On**
    - **Một điểm Off**
  - **Đối với điều kiện =, chọn:**
    - **Một điểm On**
    - **Hai điểm Off: một điểm vừa nhỏ hơn điểm On và một điểm vừa lớn hơn điểm On**

Kỹ thuật phân tích miền

Variable/ Condition Type			Test Cases														
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
X1	C11	On																
		Off																
	C12	On																
		Off																
	...	On																
		Off																
	C1m	On																
		Off																
	Typical	In																
X2	C21	On																
		Off																
	C22	On																
		Off																
	...	On																
		Off																
	C2m	On																
		Off																
	Typical	In																
Expected Result																		

Ma trận kiểm thử miền

Kỹ thuật phân tích miền

- **Ví dụ:**

```
void aFunction(int x, float y, Stack aStack)
{
  assert((y >= 1.0)          &&
         (x <= 10)           &&
         (y <= 10.0)         &&
         (x > 0)             &&
         (y <= 14.0 - x) &&
         (! aStack.isFull());
  ...
}
```

Kỹ thuật phân tích miền

Boundary			Test Cases											
Variable	Condition	Type	1	2	3	4	5	6	7	8	9	10	11	12
x	>0	On	0											
		Off		1										
	<= 10	On			10									
		Off				11								
	Typical	In					2	3	4	5	6	7	8	9
y	>= 1.00000	On					1.00000							
		Off						0.99999						
	<= 10.00000	On							10.0000					
		Off								10.0001				
	y <= 14.0 - x	On									7.00000			
		Off										7.00001		
	Typical	In	7.97320	1.78386	8.11532	6.14728							3.33333	5.11205
aStack	IsaFull( )	On											32766	
		Off												32767
	Typical	In	25	18432	4096	1	2	732	32718	9183	3718	20501		
Expected Result			✗	✓	✓	✗	✓	✗	✓	✗	✓	✗	✓	✗

Key: ✗ = IUT rejects this input, ✓ = IUT accepts this value and produces correct output, Stack values are size of stack.

FIGURE 10.21 Domain Matrix for aFunction.

Kỹ thuật bảng quyết định

- **Bảng quyết định** (*decision table*) dùng để kiểm tra, mô tả và lập tài liệu các quyết định bằng cách sử dụng một bảng.
- Xây dựng bảng quyết định:
  - Mô tả các điều kiện.
  - Xác định các giải pháp quyết định có thể có.
  - Xác định các tác vụ được thực hiện.
  - Mô tả các tác vụ.
  - Các kết quả có thể được kết hợp, đơn giản hóa.
- Bảng quyết định cho thấy cấu trúc luận lý dùng để mô tả luận lý của quá trình.

Kỹ thuật bảng quyết định

- Bảng quyết định giúp cho người phân tích bảo đảm tính đầy đủ và tính chính xác.
- Người lập trình có thể sử dụng bảng quyết định để viết mã lệnh.
- Bảng quyết định được chia thành 4 vùng:
  - Góc trên bên trái mô tả các điều kiện (*condition*).
  - Góc trên bên phải là các giá trị chọn lựa cho các điều kiện.
  - Góc dưới bên trái là các tác vụ (*action*).
  - Góc dưới bên phải là các chọn lựa tác vụ.

Kỹ thuật bảng quyết định

	Rule 1	Rule 2	…	
Conditions				
Condition 1				
Condition 2				
…				
Actions				
Action 1				
Action 2				
…				

Kỹ thuật bảng quyết định

```
for each Commission_Earned
{
  if (Extra_Bonus)
    if (Payment_Total > 50000)
    {
      add 2% to Commission_Percent;
      output Special_Letter;
      output Award_List;
    }
    else
    {
      add 1% to Commission_Percent;
      output Award_List;
    }
  else if (Payment_Total > 50000)
  {
    add 1% to Commission_Percent;
    output Special_Letter;
  }
  compute Commission = Commission_Percent * Payment_Total;
}
```

Kỹ thuật bảng quyết định

Pay Commission	Rule 1	Rule 2	Rule 3	Rule 4
Conditions				
Extra bonus	Y	Y	N	N
Payment total > 50,000 USD	Y	N	Y	N
Actions				
Add 2% to Commission Percent	X			
Add 1% to Commission Percent		X	X	
Output Special Letter	X		X	
Output Award List	X	X		

Kỹ thuật bảng quyết định

- Trong bảng quyết định:
  - Các điều kiện 1, 2, … là các điều kiện dữ liệu nhập.
  - Các hành động 1, 2, … là các hành động xảy ra tương ứng với các tổ hợp của các điều kiện dữ liệu nhập.

Kỹ thuật bảng quyết định

- **Mỗi qui tắc tương ứng một tổ hợp của các điều kiện dữ liệu nhập và thực hiện các hành động tương ứng với qui tắc này.**
  - **Các hành động không phụ thuộc vào thứ tự mà các điều kiện được định trị mà chỉ phụ thuộc vào giá trị của các điều kiện.**
  - **Các hành động chỉ phụ thuộc vào các điều kiện cụ thể, không phụ thuộc vào các điều kiện hoặc trạng thái của hệ thống trước đó.**

Kỹ thuật bảng quyết định

- Các vấn đề chính có thể xảy ra khi xây dựng bảng quyết định:
  - Tính không đầy đủ.
  - Các tình huống không thể có.
  - Các mâu thuẫn.
  - Tính dư thừa.

Kỹ thuật bảng quyết định

Conditions and Actions	Rules 1	2	3	4	5	6	7
Condition 1	Y	Y	Y	Y	Y	N	N
Condition 2	Y	Y	Y	N	N	Y	N
Condition 3	—	N	—	—	—	N	Y
Action 1	X			X	X		
Action 2		X	X			X	
Action 3							X
	Contradiction			Redundancy			

Kỹ thuật bảng quyết định

- **Qui trình kiểm thử dùng bảng quyết định**
  - **Bước 1:** Thu thập, phân tích các qui tắc nghiệp vụ và xác định các hành động được thực hiện tương ứng với mỗi qui tắc nghiệp vụ này.
  - **Bước 2:** Tạo bảng quyết định.
    - Mỗi qui tắc nghiệp vụ là một cột.
    - Mỗi điều kiện của qui tắc nghiệp vụ là một hàng.
    - Mỗi hành động là một hàng.

Kỹ thuật bảng quyết định

- **Bước 3:** Tạo các *test-case* từ các qui tắc của bảng quyết định.
  - Mỗi qui tắc có ít nhất một *test-case*.
  - Nếu các điều kiện của qui tắc là nhị phân thì tạo một *test-case* cho qui tắc này.
  - Nếu một điều kiện là một miền trị thì kiểm tra cả hai biên của miền này (kết hợp với kỹ thuật phân tích giá trị biên).

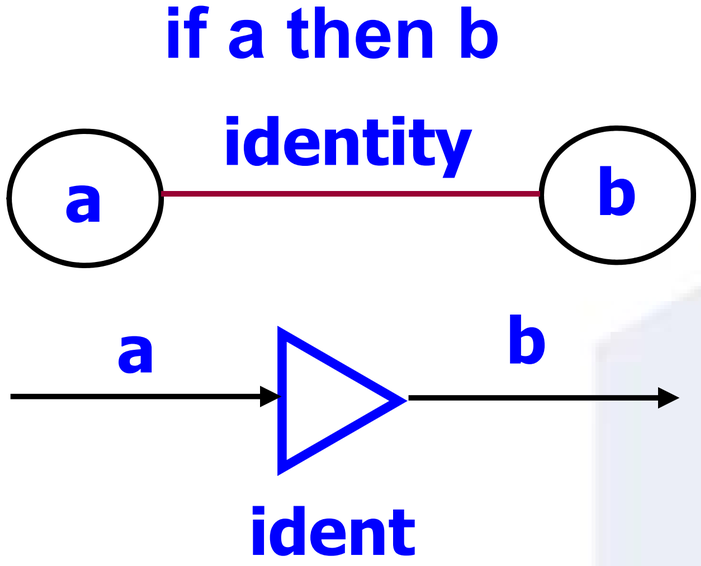
Kỹ thuật bảng quyết định

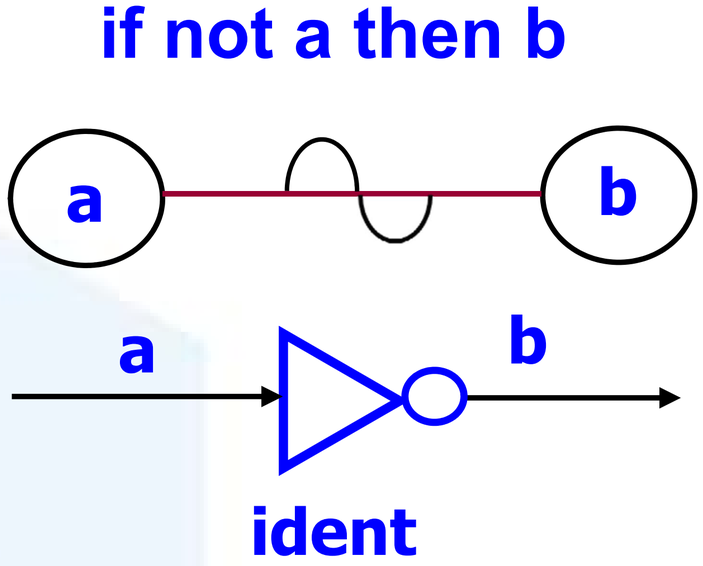
Pay Commission	Test 1	Test 2	Test 3	Test 4
Conditions				
Extra bonus	Y	Y	N	N
Payment total > 50,000 USD	51000	50000	51000	50000
Actions				
Add 2% to Commission Percent	X			
Add 1% to Commission Percent		X	X	
Output Special Letter	X		X	
Output Award List	X	X		

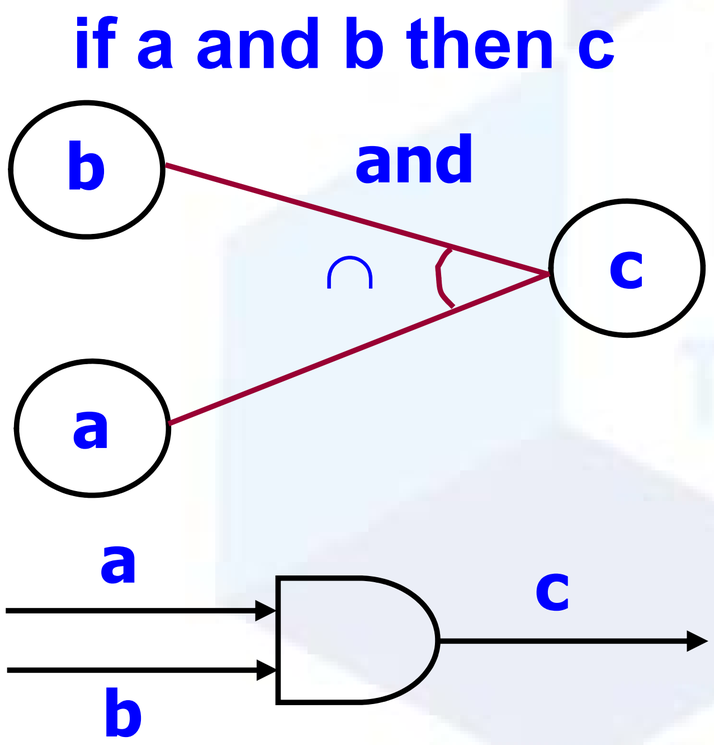
Kỹ thuật đồ thị nhân-quả

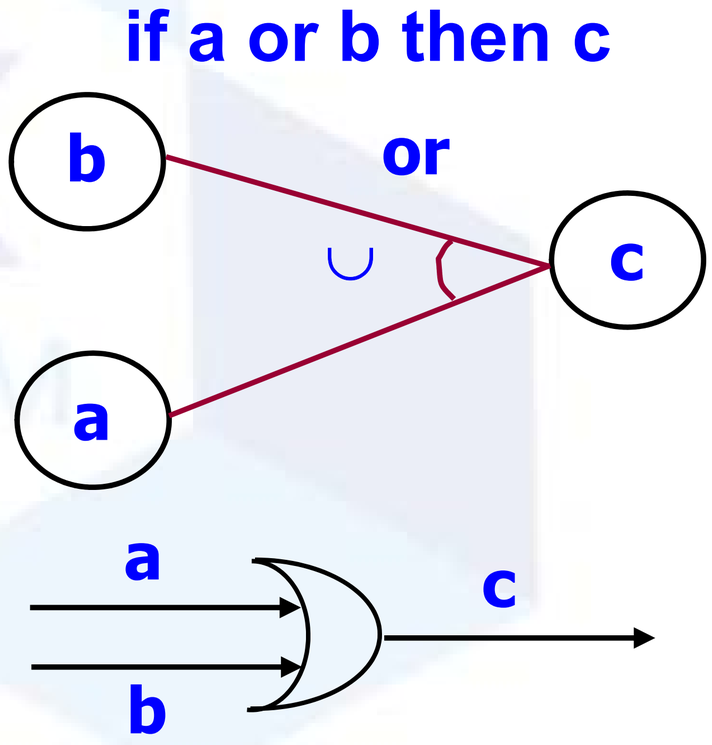
- **Đồ thị nhân-quả (*cause-effect graph*)**
  - **Đồ thị nhân-quả cho thấy các mối liên kết giữa dữ liệu nhập (*cause*) và dữ liệu xuất (*effect*).**
    - **Giải quyết các trường hợp cụ thể tương ứng với các tổ hợp của các dữ liệu nhập có thể có.**
    - **Tránh bùng nổ tổ hợp.**
  - **Các nút biểu diễn các nhân / quả.**

Kỹ thuật đồ thị nhân-quả









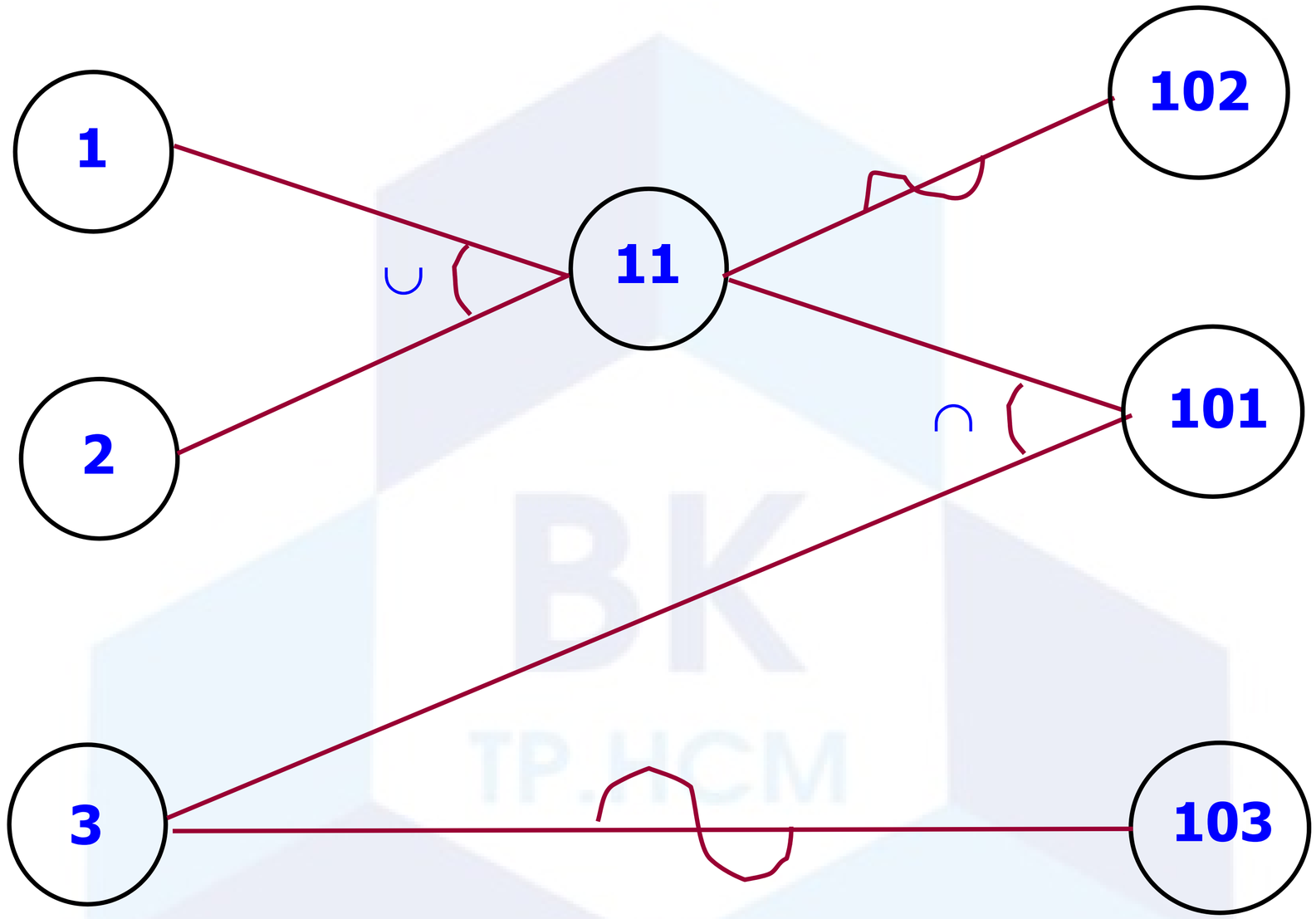
Kỹ thuật đồ thị nhân-quả

- **Ví dụ:**
  - **Dữ liệu nhập là tên tập tin gồm 2 ký tự, ký tự đầu là *A* hoặc *B*, ký tự còn lại là ký số.**
  - **Nếu ký tự đầu không phải là *A* hoặc *B* thì báo lỗi *X*1, nếu ký tự thứ 2 không phải là số thì báo lỗi *X*2.**

Kỹ thuật đồ thị nhân-quả

- **Ví dụ:**
  - **Các điều kiện dữ liệu nhập (nhân):**
    - **1: Ký tự đầu là *A***
    - **2: Ký tự đầu là *B***
    - **3: Ký tự thứ hai là ký số**
  - **Các kết quả (quả):**
    - **101: cập nhật tập tin.**
    - **102: báo lỗi *X*1.**
    - **103 : báo lỗi *X*2.**

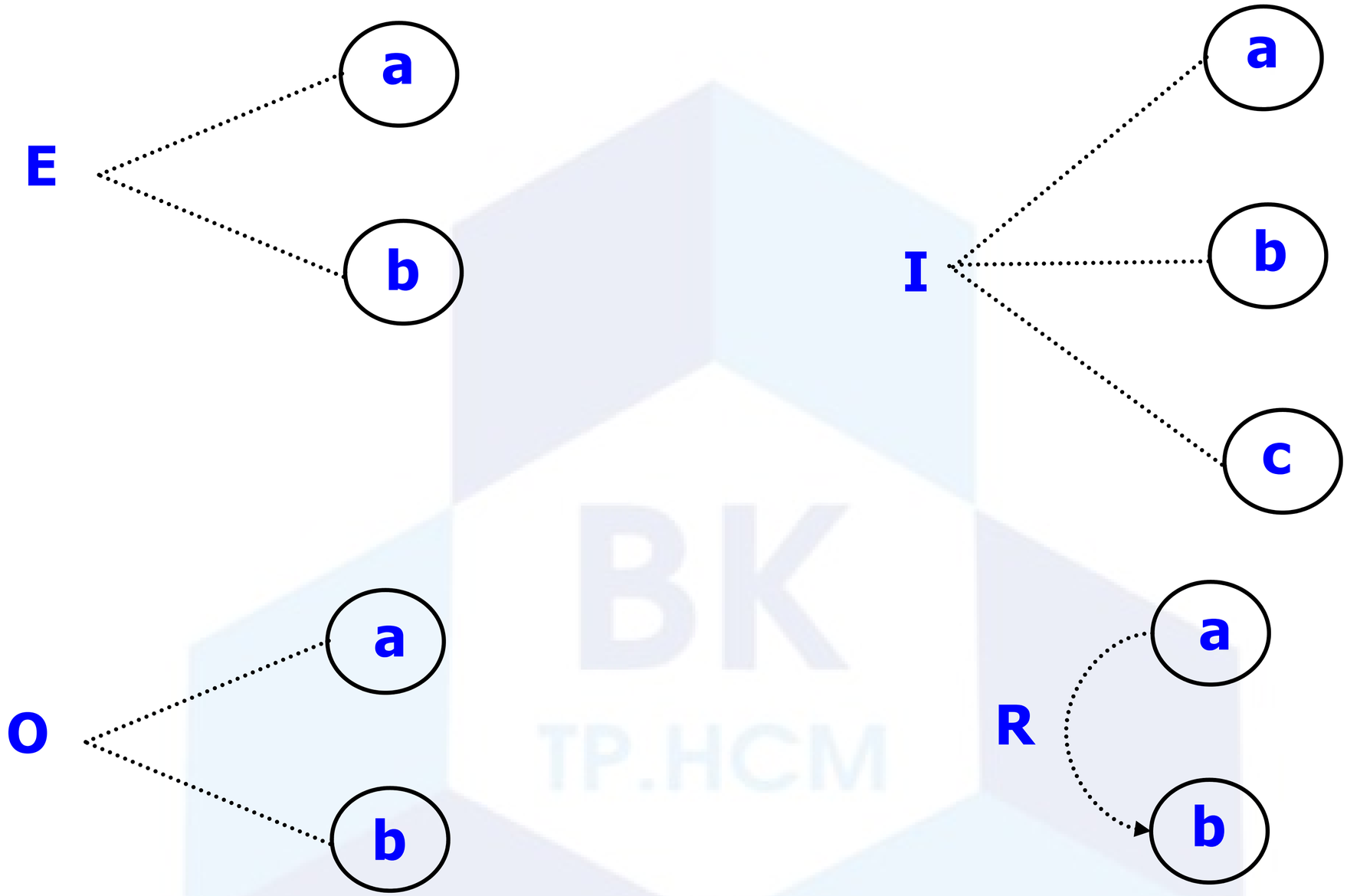
Kỹ thuật đồ thị nhân-quả



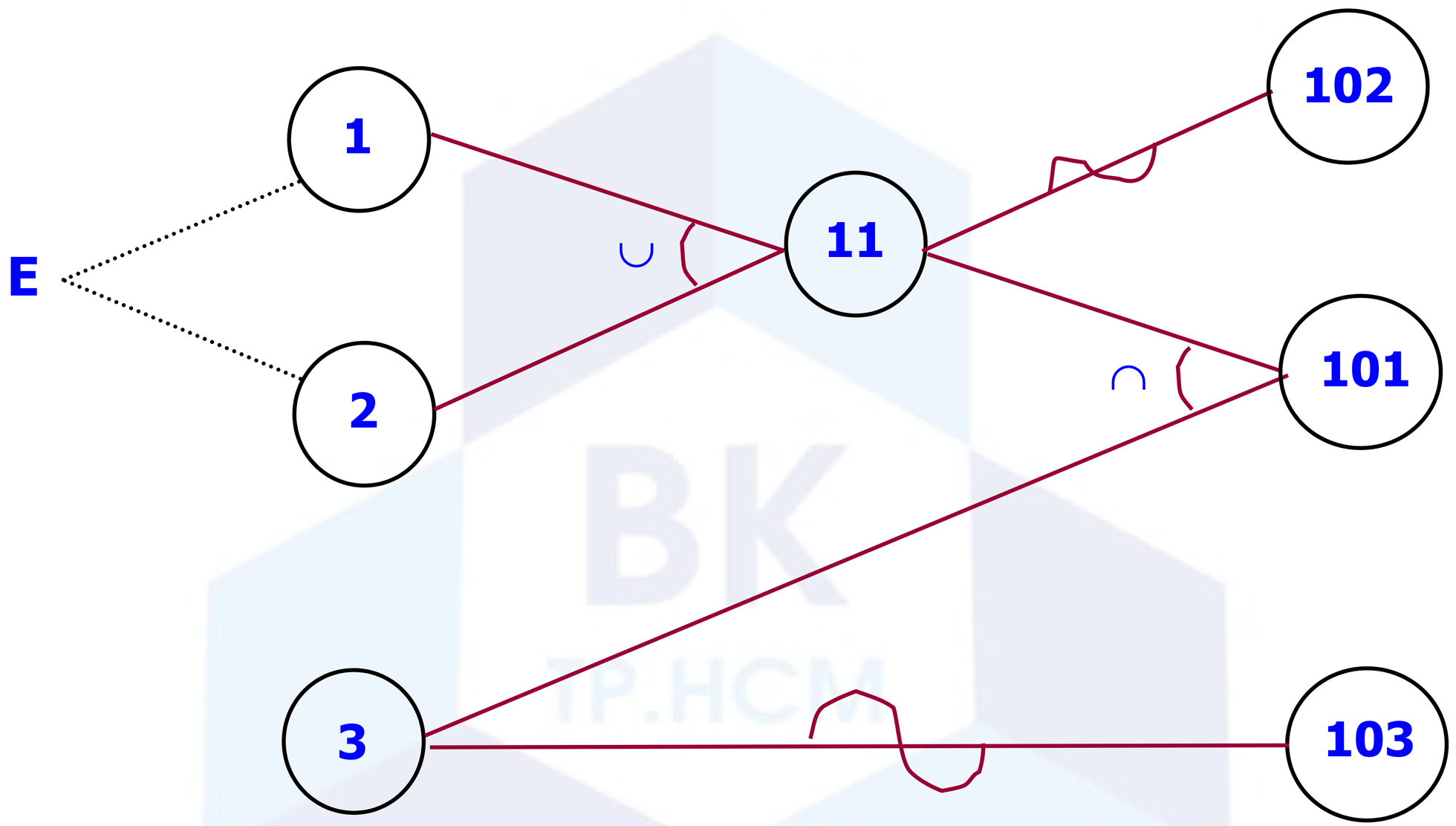
Kỹ thuật đồ thị nhân-quả

- Các ràng buộc dữ liệu nhập
  - E: không thể đồng thời xảy ra.
  - I: phải ít nhất một điều kiện xảy ra.
  - O: một và chỉ một điều kiện xảy ra.
  - R: nếu *a* xảy ra thì *b* cũng xảy ra.

Kỹ thuật đồ thị nhân-quả



Kỹ thuật đồ thị nhân-quả



Kỹ thuật đồ thị nhân-quả

- **Qui trình kiểm thử dùng đồ thị nhân-quả**
  - **Bước 1:** Xác định các dữ liệu nhập (nhân) và các hành động (quả).
  - **Bước 2:** Xây dựng đồ thị nhân-quả.
    - Xác định các ràng buộc giữa các dữ liệu nhập (nhân).
    - Xác định các mối liên kết giữa các dữ liệu nhập (nhân) và các hành động (quả).
  - **Bước 3:** Biến đổi đồ thị nhân-quả thành bảng quyết định.
  - **Bước 4:** Tạo các *test-case* từ các qui tắc của bảng quyết định.

Kỹ thuật cặp đôi

- **Kỹ thuật cặp đôi** (***pairwise testing***)

Kỹ thuật cặp đôi

- **Ví dụ: Kiểm thử một *website* với các yêu cầu:**
  - **Phải chạy tốt trên 8 trình duyệt khác nhau (*browser*): *Internet Explorer* 5.0, 5.5, 6.0, *Netscape* 6.0, 6.1, 7.0, *Mozilla* 1.1, Opera 7.**
  - **Phải chạy tốt ở 3 chế độ *plug-in*: *RealPlayer*, *MediaPlayer*, *none*.**
  - **Phải chạy tốt trên 6 hệ điều hành máy khách (*client operating system*): *Windows* 95, 98, *ME*, *NT*, 2000, *XP*.**
  - **Phải chạy tốt trên 3 *web server* khác nhau: *IIS*, *Apache*, *WebLogic*.**
  - **Phải chạy tốt trên 3 hệ điều hành máy chủ (*server operating system*): *Windows NT*, 2000, *Linux*.**

Kỹ thuật cặp đôi

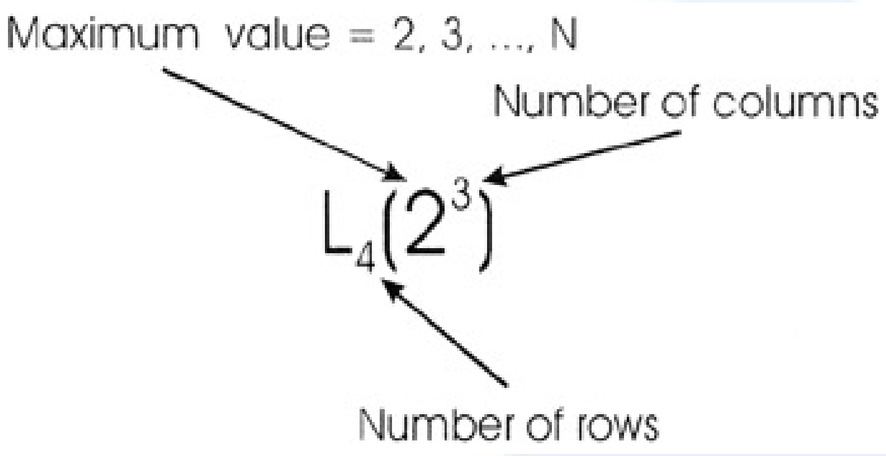
- Kiểm thử đầy đủ các yêu cầu: 8 * 3 * 6 * 3 * 3 = 1296 cấu hình khác nhau.
- **Vấn đề:**
  - Phải kiểm thử một số lượng lớn các tổ hợp.
  - Không thể có các tài nguyên để kiểm thử một số lượng lớn các tổ hợp.
  - Có thể gặp rũi ro ở một số lượng lớn các tổ hợp nếu chúng không được kiểm thử.

Kỹ thuật cặp đôi

- **Các chiến lược kiểm thử (từ xấu đến tốt)**
  - **Không kiểm thử tổ hợp nào cả.**
  - **Kiểm thử tất cả tổ hợp, trì hoản dự án, dẫn đến mất thị trường.**
  - **Kiểm thử một hoặc hai tổ hợp và hy vọng là tốt.**
  - **Kiểm thử các tổ hợp mà chúng dễ dàng được tạo ra và chạy và không quan tâm đến chất lượng.**
  - **Tạo tất cả tổ hợp và chọn một số ít các tổ hợp đầu tiên hoặc ngẫu nhiên.**
  - **Chọn một tập con đủ nhỏ các tổ hợp mà có thể phát hiện rất nhiều lỗi sai (*defect*) như mong đợi.**

Kỹ thuật cặp đôi

- **Ma trận trực giao** (***orthogonal array***) **là một bảng hai chiều gồm các số nguyên và có một đặc điểm thú vị như sau:**
  - **Hai cột bất kỳ của bảng đều có cùng các cặp đôi.**
  - **Ví dụ: hai cột bất kỳ của bảng sau có cùng các cặp đôi (1, 1), (1, 2), (2, 1) và (2, 2).**



	1	2	3
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

Kỹ thuật cặp đôi

- Ký hiệu $L_{18}(2^1,3^7)$ mô tả ma trận trực giao có 18 hàng và 8 cột, trong đó một cột chứa 2 giá trị khác nhau và 7 cột chứa 3 giá trị khác nhau.

Kỹ thuật cặp đôi

Table 6-4: $L_{18}(2^1 3^7)$ Orthogonal Array

	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1
2	1	1	2	2	2	2	2	2
3	1	1	3	3	3	3	3	3
4	1	2	1	1	2	2	3	3
5	1	2	2	2	3	3	1	1
6	1	2	3	3	1	1	2	2
7	1	3	1	2	1	3	2	3
8	1	3	2	3	2	1	3	1
9	1	3	3	1	3	2	1	2
10	2	1	1	3	3	2	2	1
11	2	1	2	1	1	3	3	2
12	2	1	3	2	2	1	1	3
13	2	2	1	2	3	1	3	2
14	2	2	2	3	1	2	1	3
15	2	2	3	1	2	3	2	1
16	2	3	1	3	2	3	1	2
17	2	3	2	1	3	1	2	3
18	2	3	3	2	1	2	3	1

Kỹ thuật cặp đôi

- **Qui trình kiểm thử dùng ma trận trực giao**
  - **Bước 1:** Xác định các biến.
  - **Bước 2:** Xác định các lựa chọn của mỗi biến.
    - Đánh số thứ tự cho các chọn lựa.
  - **Bước 3:** Xây dựng ma trận trực giao.
    - Mỗi cột tương ứng với một biến.
    - Các giá trị trong một cột tương ứng với các chọn lựa của biến.
  - **Bước 4:** Tạo các *test-case* từ các chọn lựa của ma trận trực giao.
    - Tạo *test-case* cho mỗi hàng của ma trận trực giao.

Kỹ thuật cặp đôi

- **Ví dụ:**
  - **Bước 1: Xác định các biến.**
    - ***Browser***, ***Plug-in***, ***Client operating system***, ***Web server***, ***Server operating system*** **(5 biến).**
  - **Bước 2: Xác định các lựa chọn của mỗi biến.**
    - ***Browser*****: *Internet Explorer* 5.0, 5.5, 6.0, *Netscape* 6.0, 6.1, 7.0, *Mozilla* 1.1, Opera 7 (8 chọn lựa).**
    - ***Plug-in*****: *RealPlayer*, *MediaPlayer*, *none* (3 chọn lựa).**
    - ***Client operating system*****: *Windows* 95, 98, *ME*, *NT*, 2000, *XP* (6 chọn lựa).**
    - ***Web server*****: *IIS*, *Apache*, *WebLogic* (3 chọn lựa).**
    - ***Server operating system*****: *Windows NT*, 2000, *Linux* (3 chọn lựa).**

Kỹ thuật cặp đôi

Browser
1 ↔ IE 5.0
2 ↔ IE 5.5
3 ↔ IE 6.0
4 ↔ Netscape 6.0
5 ↔ Netscape 6.1
6 ↔ Netscape 7.0
7 ↔ Mozilla 1.1
8 ↔ Opera 7

Plugin
1 ↔ None
2 ↔ RealPlayer
3 ↔ MediaPlayer
4 ↔ Not used (at this time)

Client operating system
1 ↔ Windows 95
2 ↔ Windows 98
3 ↔ Windows ME
4 ↔ Windows NT
5 ↔ Windows 2000
6 ↔ Windows XP
7 ↔ Not used (at this time)
8 ↔ Not used (at this time)

Server
1 ↔ IIS
2 ↔ Apache
3 ↔ WebLogic
4 ↔ Not used (at this time)

Server operating system
1 ↔ Windows NT
2 ↔ Windows 2000
3 ↔ Linux
4 ↔ Not used (at this time)

Kỹ thuật cặp đôi

- **Ví dụ:**
  - **Bước 3: Xây dựng ma trận trực giao.**
    - **Kích thước của ma trận trực giao (5 cột): $8^1 6^1 3^3$**
    - **Chọn ma trận trực giao: $L_{64}(8^2 4^3)$**

Kỹ thuật cặp đôi

$L_{64}(8^2 4^3)$ Orthogonal Array

	1	2	3	4	5
1	1	1	1	1	1
2	1	4	3	4	4
3	1	4	2	4	4
4	1	1	4	1	1
5	1	3	5	3	3
6	1	2	7	2	2
7	1	2	6	2	2
8	1	3	8	3	3
9	3	4	1	3	3
10	3	1	3	2	2
11	3	1	2	2	2

~

57	8	1	1	1	4
58	8	4	3	4	1
59	8	4	2	4	1
60	8	1	4	1	4
61	8	3	5	3	2
62	8	2	7	2	3
63	8	2	6	2	3
64	8	3	8	3	2

Kỹ thuật cặp đôi

$L_{64}(8^2 4^3)$ with a full mapping of all its columns.

	Browser	Plug-in	Client OS	Server	Server OS
1	IE 5.0	None	Win 95	IIS	Win NT
2	IE 5.0	4	Win ME	4	4
3	IE 5.0	4	Win 98	4	4
4	IE 5.0	None	Win NT	IIS	Win NT
5	IE 5.0	MediaPlayer	Win 2000	WebLogic	Linux
6	IE 5.0	RealPlayer	7	Apache	Win 2000
7	IE 5.0	RealPlayer	Win XP	Apache	Win 2000
8	IE 5.0	MediaPlayer	8	WebLogic	Linux
9	IE 6.0	4	Win 95	WebLogic	Linux
10	IE 6.0	None	Win ME	Apache	Win 2000
11	IE 6.0	None	Win 98	Apache	Win 2000

~

57	Opera 7	None	Win 95	IIS	4
58	Opera 7	4	Win ME	4	Win NT
59	Opera 7	4	Win 98	4	Win NT
60	Opera 7	None	Win NT	IIS	4
61	Opera 7	MediaPlayer	Win 2000	WebLogic	Win 2000
62	Opera 7	RealPlayer	7	Apache	Linux
63	Opera 7	RealPlayer	Win XP	Apache	Linux
64	Opera 7	MediaPlayer	8	WebLogic	Win 2000

Kỹ thuật cặp đôi

L_{64} ($8^2 4^3$) with a full mapping of all its columns including the "extra" cells.

	Browser	Plug-in	Client OS	Server	Server OS
1	IE 5.0	None	Win 95	IIS	Win NT
2	IE 5.0	None	Win ME	IIS	Win NT
3	IE 5.0	None	Win 98	IIS	Win NT
4	IE 5.0	None	Win NT	IIS	Win NT
5	IE 5.0	MediaPlayer	Win 2000	WebLogic	Linux
6	IE 5.0	RealPlayer	Win 95	Apache	Win 2000
7	IE 5.0	RealPlayer	Win XP	Apache	Win 2000
8	IE 5.0	MediaPlayer	Win 98	WebLogic	Linux
9	IE 6.0	None	Win 95	WebLogic	Linux
10	IE 6.0	None	Win ME	Apache	Win 2000
11	IE 6.0	None	Win 98	Apache	Win 2000

~

57	Opera 7	None	Win 95	IIS	Win NT
58	Opera 7	None	Win ME	IIS	Win NT
59	Opera 7	None	Win 98	IIS	Win NT
60	Opera 7	None	Win NT	IIS	Win NT
61	Opera 7	MediaPlayer	Win 2000	WebLogic	Win 2000
62	Opera 7	RealPlayer	Win 95	Apache	Linux
63	Opera 7	RealPlayer	Win XP	Apache	Linux
64	Opera 7	MediaPlayer	Win 98	WebLogic	Win 2000

Kỹ thuật sơ đồ chuyển trạng thái

- **Sơ đồ chuyển trạng thái** (***state transition diagram***) **dùng để đặc tả hành vi (*behavior*) của hệ thống. Hành vi này được phân chia thành các chuỗi sự kiện (*event*) và có thể gây ra một hoặc nhiều trạng thái có thể có.**
  - **Sơ đồ chuyển trạng thái biểu diễn các trạng thái của các đối tượng của một lớp và cho thấy sự thay đổi các trạng thái khi có các sự kiện xảy ra.**

Kỹ thuật sơ đồ chuyển trạng thái

❖ Các thành phần của sơ đồ chuyển trạng thái

▸ Trạng thái (*state*)

- **Trạng thái là tình trạng mà hệ thống đang chờ một hoặc nhiều sự kiện xảy ra.**
- **Sự kiện gây ra sự chuyển trạng thái và/hoặc khởi động các hành động.**
- **Trạng thái được biểu diễn bởi các giá trị của một hoặc nhiều biến trong hệ thống.**
- **Trạng thái được biểu diễn bởi một hình tròn bên trong ghi tên trạng thái (danh từ số ít).**

▸ Chuyển trạng thái (*transition*)

- **Chuyển trạng thái biểu diễn sự thay đổi từ một trạng thái sang một trạng thái khác do một sự kiện gây ra.**

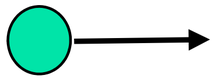
Kỹ thuật sơ đồ chuyển trạng thái

- **Sự kiện (*event*)**
  - **Sự kiện là sự việc xảy ra và làm cho hệ thống bị thay đổi trạng thái.**
  - **Sự kiện được đưa vào hệ thống thông qua các giao tiếp của hệ thống, hoặc được tạo ra ở bên trong hệ thống.**
  - **Các sự kiên có thể liên quan với nhau hoặc độc lập nhau.**
  - **Khi một sự kiện xảy ra, hệ thống có thể thay đổi trạng thái hoặc giữ nguyên trạng thái và/hoặc thực hiện một hành động.**
  - **Sự kiện có thể có các tham số kèm theo.**
  - **Sự kiện được biểu diễn bởi một tên là động từ ghi bên cạnh mũi tên chuyển trạng thái.**

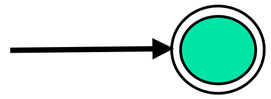
Kỹ thuật sơ đồ chuyển trạng thái

- **Hành động (*action*)**
  - **Hành động là một hoạt động xảy ra khi chuyển trạng thái.**
  - **Hành động làm cho hệ thống phải thực hiện một công việc nào đó hoặc tạo ra các kết xuất.**
  - **Hành động được biểu diễn bởi một lệnh đi sau dấu '/'.**

Kỹ thuật sơ đồ chuyển trạng thái



Điểm bắt đầu



Điểm kết thúc

Trạng thái

Sự kiện

Hành động

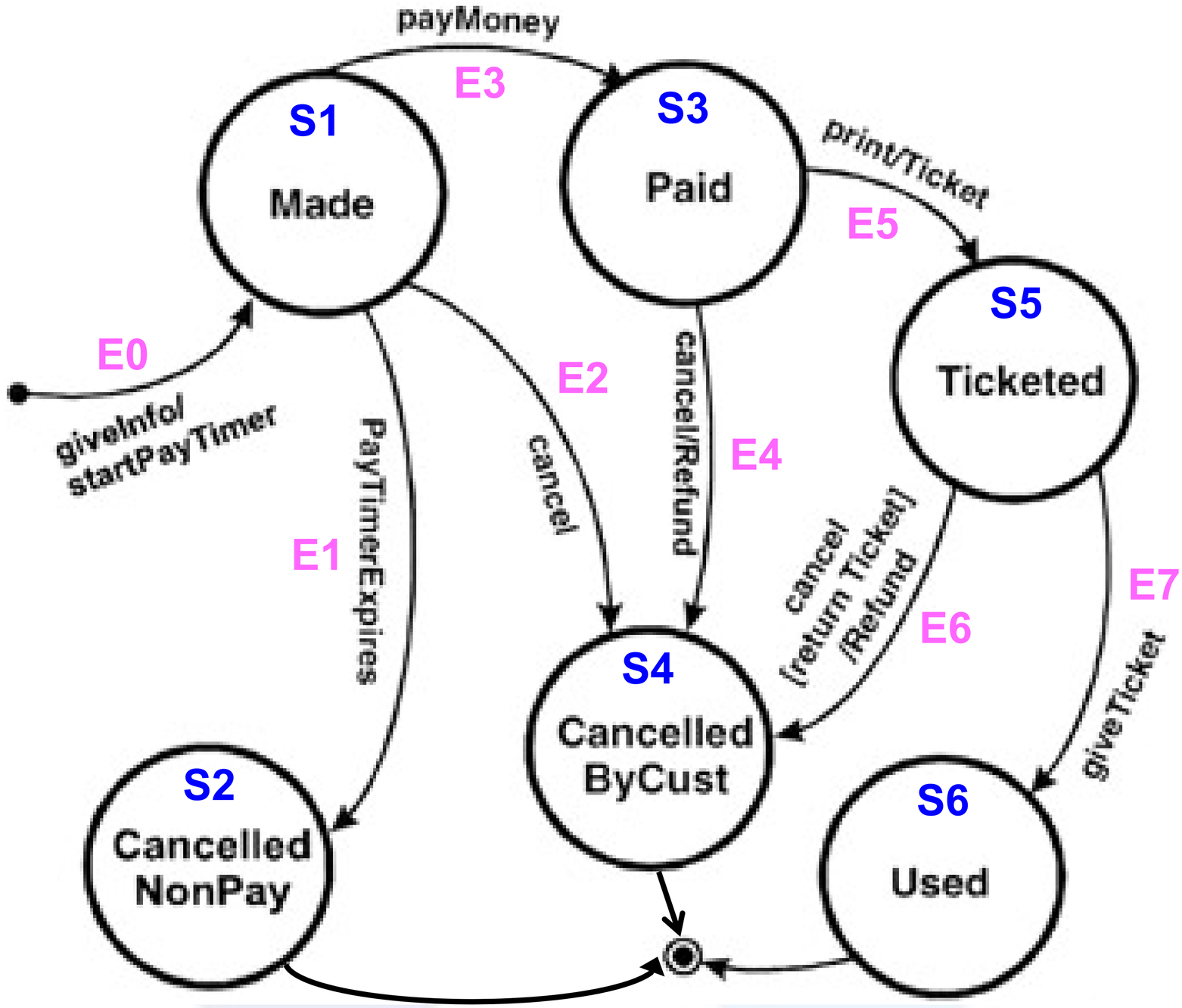
Kỹ thuật sơ đồ chuyển trạng thái

- **Ví dụ: Hệ thống đặt mua vé máy bay có 6 trạng thái khác nhau:**
  - ***Made* (S1)**
    - **Sự kiện dẫn đến: Bắt đầu nhập thông tin khách hàng (*giveInfo*).**
    - **Hành động kèm theo: Khởi động thời hạn trả tiền *PayTimer*.**
  - ***CancelledNonPay* (S2)**
    - **Sự kiện dẫn đến: Quá thời hạn trả tiền *PayTimer* (*PayTimerExpires*).**
  - ***Paid* (S3)**
    - **Sự kiện dẫn đến: Khách hàng đã trả tiền.**

Kỹ thuật sơ đồ chuyển trạng thái

- ***CancelledByCust*** **(S4)**
  - **Sự kiện dẫn đến: Khách hàng hủy bỏ đặt vé, hoặc hủy bỏ đặt vé và hoàn tiền, hoặc hoàn vé và hoàn tiền.**
- ***Ticketed*** **(S5)**
  - **Sự kiện dẫn đến: In vé.**
  - **Hành động kèm theo: vé.**
- ***Used*** **(S6)**
  - **Sự kiện dẫn đến: Khách hàng đã sử dụng vé.**

Kỹ thuật sơ đồ chuyển trạng thái



Kỹ thuật sơ đồ chuyển trạng thái

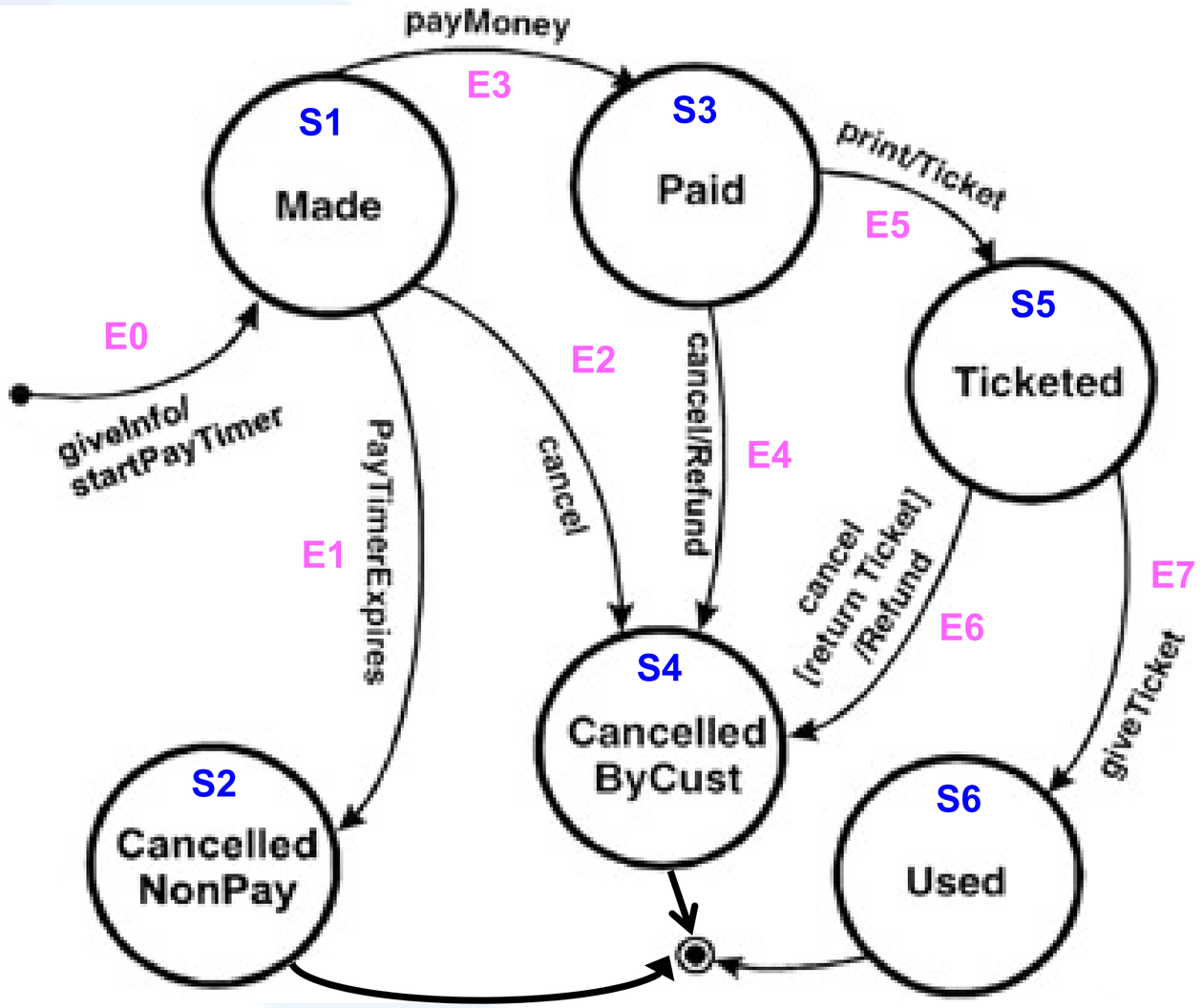
Trạng thái hiện tại	Sự kiện	Hành động/Kết quả	Trạng thái kế tiếp
Bắt đầu	Nhập thông tin khách hàng (givelnfo)	Khởi động thời hạn trả tiền PayTimer	Made
Made	Khách hàng đã trả tiền		Paid
Made	Khách hàng hủy bỏ đặt vé		Cancelled ByCust
Made	Quá thời hạn trả tiền PayTimer		Concelled NonPay
Paid	In vé	Vé	Tiketed
Paid	Hủy bỏ đặt vé	Hoàn tiền	Concelled ByCust
Ticketed	Hoàn vé	Hoàn tiền	Concelled ByCust
Ticketed	Khách hàng sử dụng vé		Used

Bảng chuyển trạng thái

Kỹ thuật sơ đồ chuyển trạng thái

- **Mức phủ trạng thái:** Duyệt *DFS* để tìm các đường cơ bản có ít nhất một trạng thái chưa duyệt.

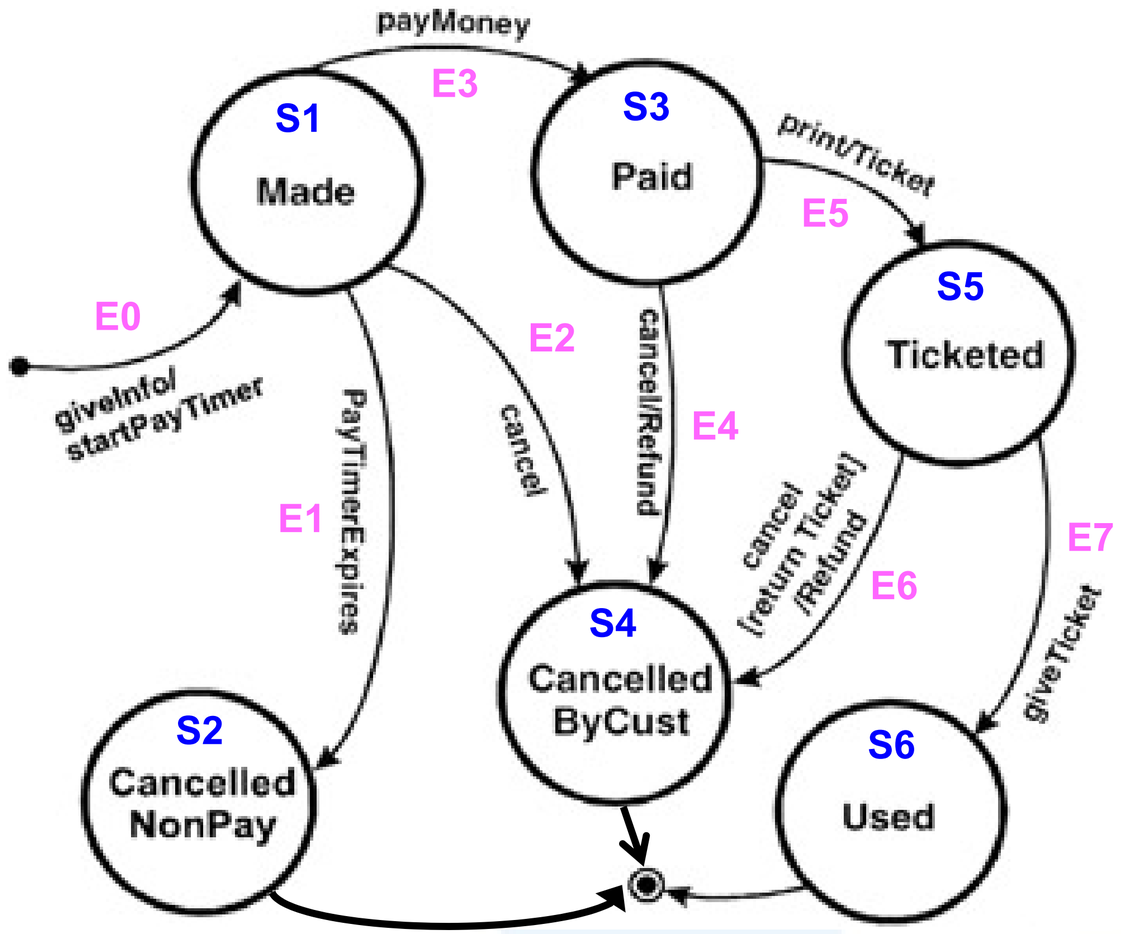
P1: Made – CancelledNonPay
P2: Made – CancelledByCust
P3: Made – Paid – Ticketed – Used



Kỹ thuật sơ đồ chuyển trạng thái

- **Mức phủ chuyển trạng thái:** Lập bảng chuyển trạng thái.

Kỹ thuật sơ đồ chuyển trạng thái



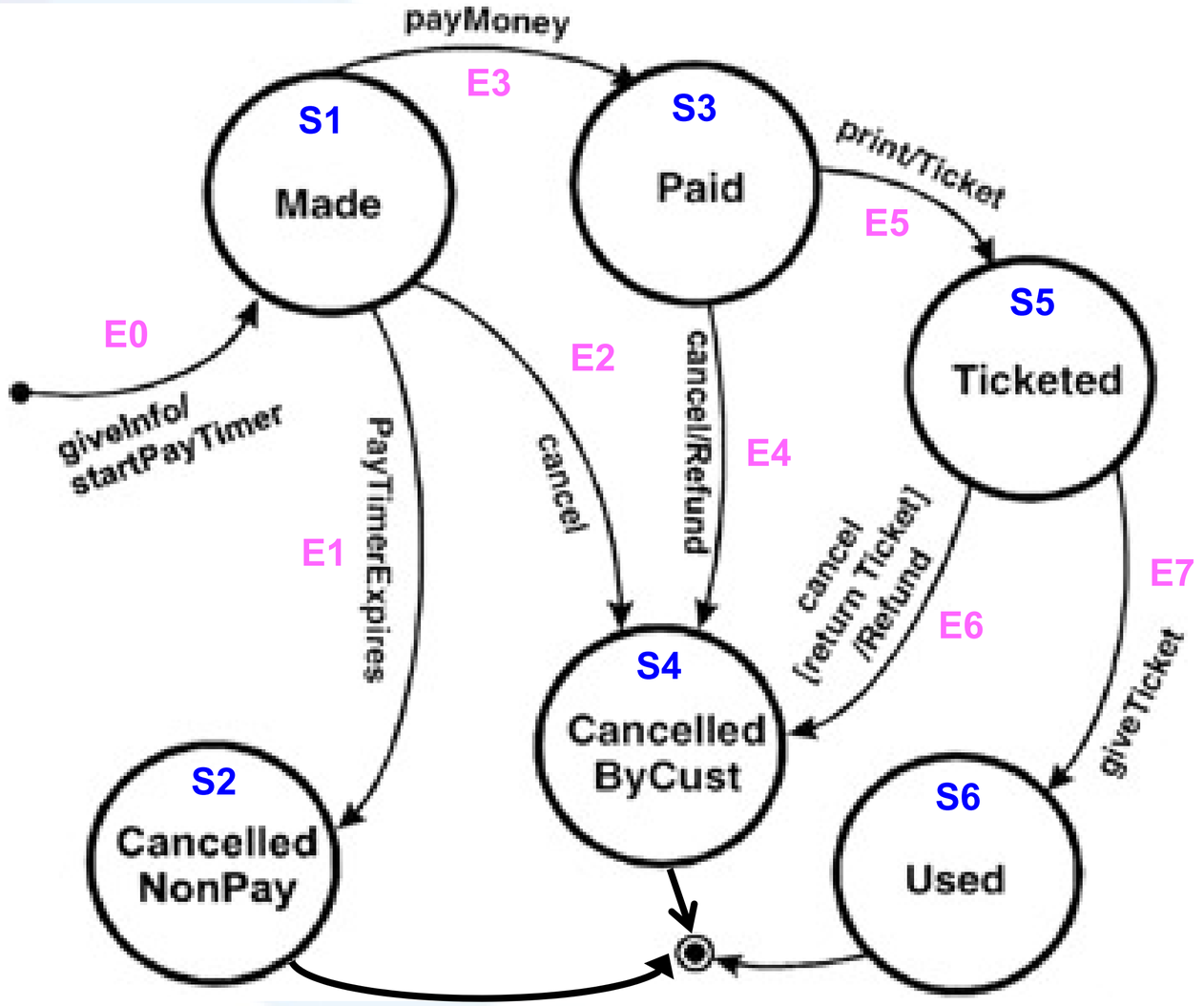
Bảng chuyển trạng thái

Test-case	TC1	TC2	TC3	TC4	TC5	TC6	TC7
Start state	S1	S1	S1	S3	S3	S5	S5
Input	E1	E3	E2	E4	E5	E6	E7
Output				Refund	Ticket	Return Ticket Refund	
Finish state	S2	S3	S4	S4	S5	S4	S6

Kỹ thuật sơ đồ chuyển trạng thái

- **Mức phủ sự kiện:** Duyệt *DFS* để tìm các đường cơ bản có ít nhất một sự kiện chưa duyệt.

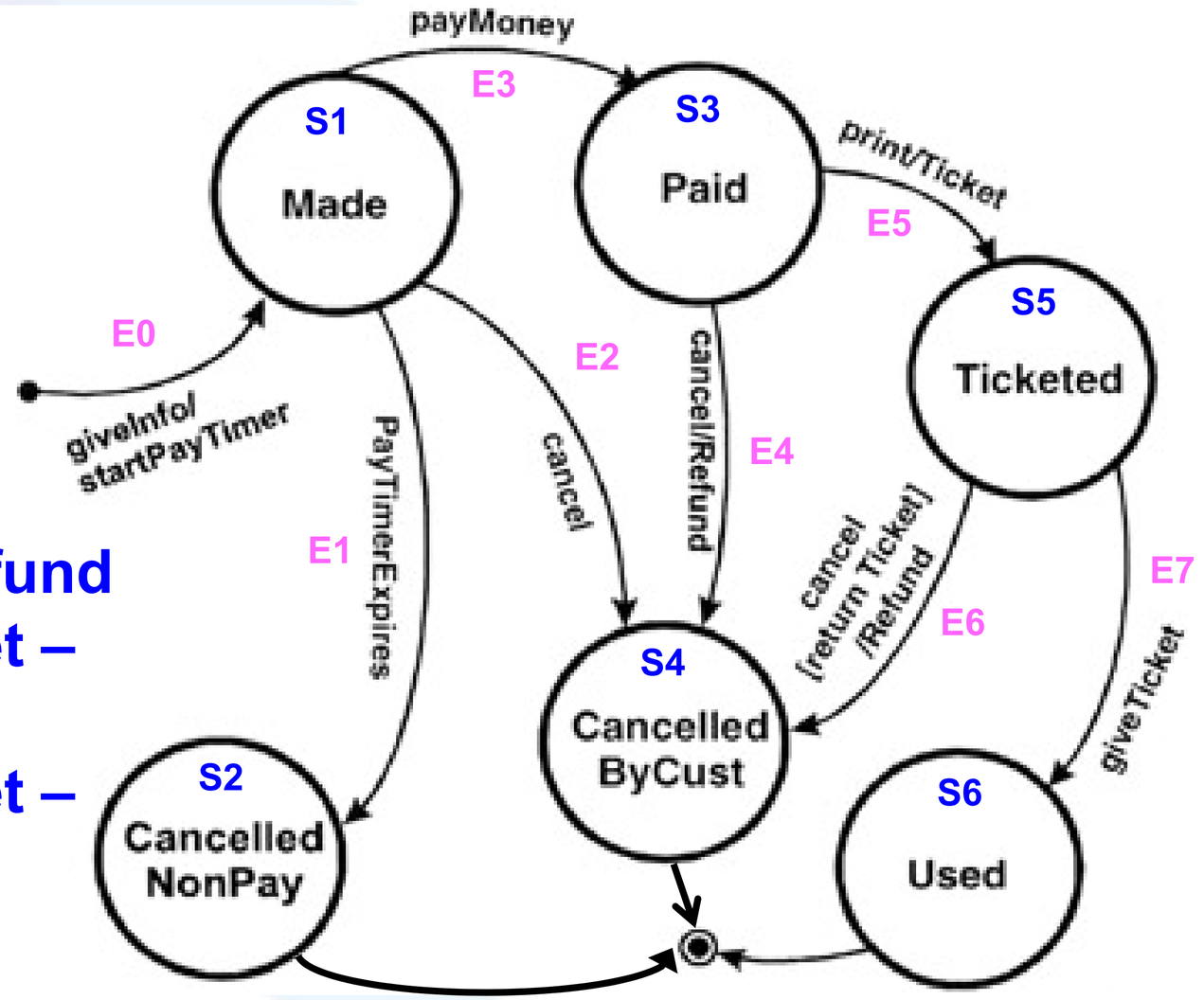
P1: giveInfo – payTimerExpires
P2: giveInfo – cancel
P3: giveInfo – payMoney – print – giveTicket



Kỹ thuật sơ đồ chuyển trạng thái

- **Mức phủ đường cơ bản:** Duyệt *DFS* để tìm tất cả các đường cơ bản. Đây là mức phủ tốt nhất.

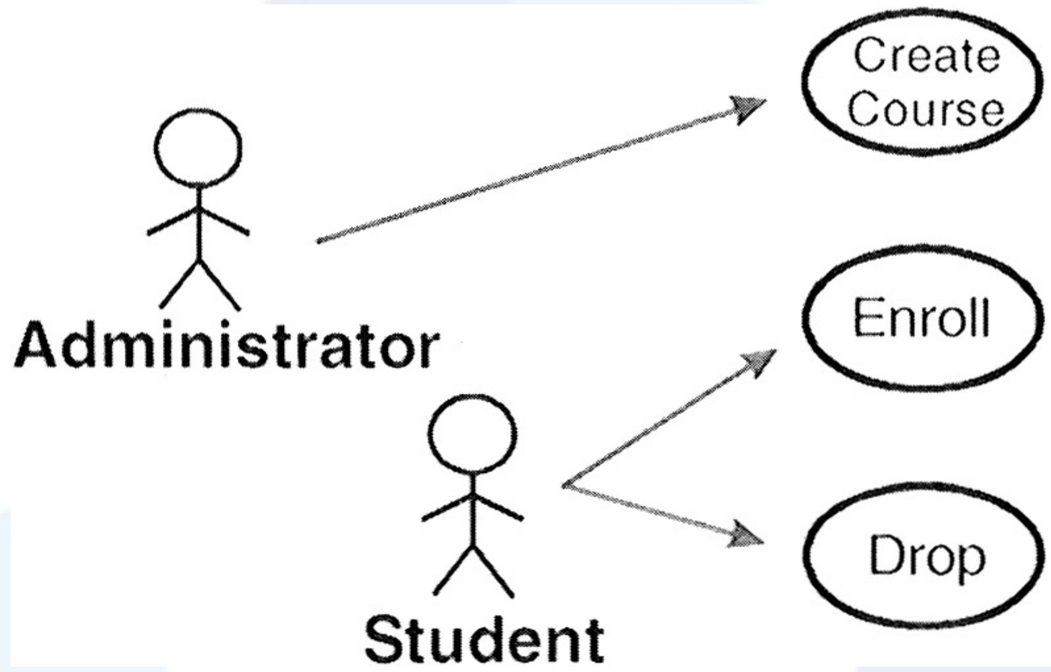
P1: giveInfo – payTimerExpires
P2: giveInfo – cancel
P3: giveInfo – payMoney – cancel/Refund
P4: giveInfo – payMoney – print/Ticket – cancel[Return Ticket]/Refund
P5: giveInfo – payMoney – print/Ticket – giveTicket



Kỹ thuật *use-case*

- **Kỹ thuật *use-case* (*use-case testing*) xác định các *test-case* chạy trên toàn bộ hệ thống đi từ giao tác này đến giao tác khác từ lúc bắt đầu cho đến lúc kết thúc.**
  - ***Use-case* là một đặc tả trường hợp cụ thể mà tác nhân (*actor*) sử dụng hệ thống.**
  - ***Use-case* mô tả các giao tiếp của tác nhân với hệ thống để thực hiện một công việc cụ thể**
  - ***Use-case* là một chuỗi các bước (kịch bản) mô tả các giao tiếp mà tác nhân phải làm với hệ thống.**
    - **Những gì nhập vào hệ thống (*input*).**
    - **Những gì mà hệ thống cho ra (*output*).**

Kỹ thuật *use-case*



Kỹ thuật *use-case*

Table 9-1: Use case template.

Use Case Component	Description	
Use Case Number or Identifier	A unique identifier for this use case	
Use Case Name	The name should be the goal stated as a short active verb phrase	
Goal in Context	A more detailed statement of the goal if necessary	
Scope	Corporate | System | Subsystem	
Level	Summary | Primary task | Subfunction	
Primary Actor	Role name or description of the primary actor	
Preconditions	The required state of the system before the use case is triggered	
Success End Conditions	The state of the system upon successful completion of this use case	
Failed End Conditions	The state of the system if the use case cannot execute to completion	
Trigger	The action that initiates the execution of the use case	
Main Success Scenario	**Step**	**Action**
	1	
	2	
	...	
Extensions	Conditions under which the main success scenario will vary and a description of those variations	
Sub-Variations	Variations that do not affect the main flow but that must be considered	
Priority	Criticality	
Response Time	Time available to execute this use case	
Frequency	How often this use case is executed	
Channels to Primary Actor	Interactive | File | Database | ...	
Secondary Actors	Other actors needed to accomplish this use case	
Channels to Secondary Actors	Interactive | File | Database | ...	
Date Due	Schedule information	
Completeness Level	Use Case identified (0.1)| Main scenario defined (0.5) | All extensions defined (0.8) | All fields complete (1.0)	
Open Issues	Unresolved issues awaiting decisions	

Use Case Component	Description	
Use Case Number or Identifier	SURS1138	
Use Case Name	Register for a course (a class taught by a faculty member)	
Goal in Context		
Scope	System	
Level	Primary task	
Primary Actor	Student	
Preconditions	None	
Success End Conditions	The student is registered for the course—the course has been added to the student's course list	
Failed End Conditions	The student's course list is unchanged	
Trigger	Student selects a course and "Registers"	
Main Success Scenario A: Actor S: System	Step	Action
	1	A: Selects "Register for a course"
	2	A: Selects course (e.g. Math 1060)
	3	S: Displays course description
	4	A: Selects section (Mon & Wed 9:00am)
	5	S: Displays section days and times
	6	A: Accepts
	7	S: Adds course/section to student's course list
Extensions	2a	Course does not exist S: Display message and exit
	4a	Section does not exist S: Display message and exit
	4b	Section is full S: Display message and exit
	6a	Student does not accept S: Display message and exit
Extensions	2a	Course does not exist S: Display message and exit
	4a	Section does not exist S: Display message and exit
	4b	Section is full S: Display message and exit
	6a	Student does not accept S: Display message and exit
Sub-Variations	Student may use ▪ Web ▪ Phone	
Priority	Critical	
Response Time	10 seconds or less	
Frequency	Approximately 5 courses x 10,000 students over a 4-week period	
Channels to Primary Actor	Interactive	
Secondary Actors	None	
Channels to Secondary Actors	N/A	
Date Due	1 Feb	
Completeness Level	0.5	
Open Issues	None	

Kỹ thuật *use-case*

- **Kiểm thử dùng *use-case*:**
  - **Kiểm thử tất cả các *use-case*.**
  - **Ít nhất một *test-case* để kiểm thử kịch bản chính chính.**
  - **Ít nhất một *test-case* để kiểm thử mỗi phần mở rộng.**
- **Lưu ý:**
  - **Vì *use-case* không có đặc tả dữ liệu nhập (*input*), do đó người kiểm thử (*tester*) phải xác định dữ liệu nhập bằng các kỹ thuật phân chia lớp tương đương, phân tích giá trị biên, phân tích miền.**